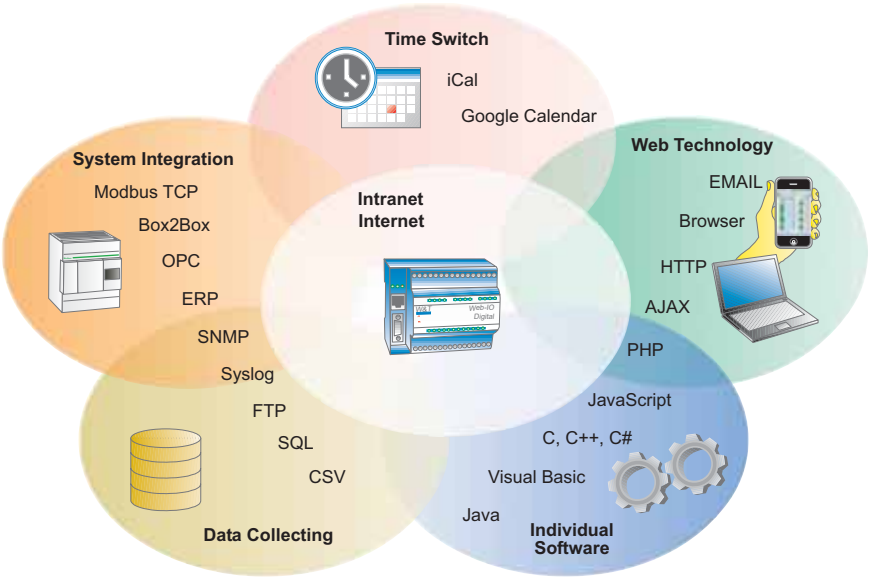
Manual
Web-IO Digital



Type:

#57637 Web-IO 2xDigital In, 2xDigital Out
#57630M Web-IO 12xDigital In, 12xDigital Out
#57631M Web-IO 12xDigital In, 12xDigital Out with Com-Server
#57634M Web-IO 12xDigital In, 6xRelaisOut
#57638 ERP-Gateway 2xDig. In, 2xDig. Out

Release 3.25 April 2011

Firmware 3.26

W&T

© 04/2011 by Wiesemann und Theis GmbH
Microsoft, MS-DOS, Windows, Winsock and Visual Basic
are registered trademarks of Microsoft Corporation

Subject to error and alteration:

Since it is possible that we make mistakes, you mustn't use any of our statements without verification. Please, inform us of any error or misunderstanding you come about, so we can identify and eliminate it as soon as possible.

Carry out your work on or with W&T products only to the extent that they are described here and after you have completely read and understood the manual or guide. We are not liable for unauthorized repairs or tampering. When in doubt, check first with us or with your dealer.

Content

1. Overview of Potential Uses 9
1.1 Overview of potential uses 10
1.1.1 Browser access 10
For PC browsers 10
For smart phone 10
Your own Web pages 10
1.1.2 Box-to-Box 10
1.1.3 Time-based switching 10
Use of a local calendar program 11
Scheduling switching times in Google Calendar 11
1.1.4 Integration into existing systems 11
OPC 11
Modbus TCP 11
SNMP - Simple Network Management Protocol 11
1.1.5 Access from your own applications 11
Communication via HTTP-Requests 12
Communication via command string 12
Communication via binary structures 12
1.1.6 Email and more - information when status changes 12
1.1.7 Acquiring switching states 13
Messages via the Web-IO alarm system 13
Acquiring IO states in the W&T Motherbox 13
1.1.8 Logic functions - linking inputs and outputs 13
1.1.9 Basic port behavior 13
Inputs 13
Outputs 14
1.1.10 General basic functions 14

2 Before starting up 15
2.1 Supply voltage and network connection 16
2.1.1 #57637 and #57638 16
2.2 Assigning the IP address 18
2.2.1 Automatically obtaining the IP address via DHCP 18
2.2.2 Assigning the address using the Wutility-Tool 18

3 Connections, operating elements and LEDs 21
3.1 Connecting Web-IO 2xDig. In, 2xDig. Out PoE 22
3.1.1 Supply voltage input (Vcc) 22

3.1.2 Input wiring 23
3.1.3 Output circuit 25
3.2 Connecting Web-IO 12xDig. In, 12xDig. Out 28
3.2.1 Supply voltage input 28
3.2.2 Input circuit 29
3.3.3 Output circuit 31
3.4 Connecting Web-IO 12xDig. In, 6xDig. Relais 33
3.4.1 Supply voltage input 33
3.4.2 Input circuit 33
3.4.3 Output circuit 35
3.6 Ethernet connection 37
3.7 Serial connection 39
3.8 LEDs for device status 40

4 Configuring the network parameters 43
4.1 Web-Based Management 44
4.1.1 Structure of the Web site 44
4.1.2 Navigation 44
4.1.3 Easier navigation with profiles 44
4.1.4 Access rights 45
4.1.5 Changing and creating the access passwords 46
4.1.6 Saving and activating settings 47
4.2 Basic setting of the network parameters 49

5 Basic Functions 55
5.1 HTTP - Controlling IOs in the browser 56
5.1.1 The Home page 56
5.1.2 The User page 57
5.1.3 The Smart page 59
5.1.4 Labeling and texts 60
5.1.5 Changing the HTTP port 64
5.2 Box-to-Box 66
5.2.1 Configuring the Slave Web-IO 66
5.2.2 Configuring the master 70
5.2.3 GetBox-to-Box connection status 74
5.2.4 Stopping Box-to-Box mode 74
5.2.6 Box-to-Box notes and limitations 77
5.3 Time Switch - Using the Web-IO as a Time Switch 80
5.3.1 Activate the outputs for timed switching 80
5.3.2 Load the switching defaults 81
5.3.3 Load switching times from an online calendar 81

5.3.4	Load switching times as a iCal file	87
5.3.5	Time Switch preview	91
5.3.6	Special features of the Web-IO Time Switch	92
5.4	**Integration into existing systems**	**93**
5.4.1	SNMP - In-/Outputs in management systems	93
	Enabling SNMP access	93
	Activating the outputs for SNMP	95
	Main steps for SNMP access	96
5.4.2	OPC - Standardized access	102
	Installing the OPC server	102
	Uninstalling	102
	Configuration	102
	Configuring the Web-IO as an OPC device	105
5.4.3	Modbus TCP - standardized access	109
	Modbus TCP communication	111
	Function Code 0x01 Read Coils	114
	Function Code 0x02 Read Discrete Inputs	115
	Function Code 0x03 Read Holding Registers	116
	Function Code 0x04 Read Input Registers	118
	Function Code 0x05 Write Single Coil	118
	Function Code 0x06 Write Single Register	119
	Function Code 0x0F Write Multiple Coils	120
	Function Code 0x10 Write Multiple Registers	122
	Function Code 0x07 Read Exception State	123
	Incorrect master requests	124
	Modbus address areas for Web-IO Digital	125
	Modbus - virtual registers	127
5.5	**Email and more - Alarms**	**132**
5.5.1	Determining the alarm conditions	132
5.5.2	E-Mail-Alarm - SMTP	137
5.5.3	SNMP Alarm	141
5.5.4	Alarm via UDP	143
	Switching outputs for a different Web-IO using UDP	145
5.5.5	Alarm message to your own TCP applications	146
5.5.6	Alarm message to a TCP server	147
	Switching outputs for a different Web-IO using TCP	149
5.5.7	Sending alarm messages to a Syslog-Server	150
5.5.8	Sending alarms via FTP Server	153
5.5.9	General information about alarms	156
5.6	**Access from your own application**	**157**
5.6.1	Access from the browser	157

	Using W&T Tags	158
	Dynamic Web pages with JavaScript and AJAX	163
	Using the Web-IO's own Java applets	172
	Using the applet with JavaScript	175
5.6.2	Access from various high level languages	179
	Socket programming with command strings	180
	TCP communication - the Web-IO as server	183
	TCP communication - the Web-IO as Client	190
	UDP communication	196
	Program-controlled error management	200
	BINARY - socket programs with binary structures	206
	Specifying the operation mode	207
	The Web-IO as socket server	208
	The Web-IO as socket client	210
	The Web-IO as UDP Peer	213
	Passwort protection	216
	BINARY - The IO structures	218
	Definition of the IO structures	219
	Working with the IO structures	220
5.7	**Wayback - switching via FTP login**	**229**
5.7.1	Functional principle	229
5.7.2	The Wayback commands	230
5.7.3	Enabling Wayback access	231
5.7.4	Activating the outputs for Wayback access	233
6	**General Functions**	**235**
6.1	**Time / Date - Setting**	**236**
6.1.1	Time zones	236
6.1.2	Summer time / Winter time	237
6.1.3	Start and end of summer time	237
6.1.4	End of summer time	238
6.1.5	Time-Server	239
6.1.6	Manually setting time and Date	241
6.1.7	Activating all time/date settings	242
6.2	**Safety Timeout / State - Access monitoring**	**243**
6.2.1	Specifying safety timeout	243
6.2.2	Specify Safety State	244
6.3	**Ports - Input- and output configuration**	**246**
6.3.1	Input configuration	246
6.3.2	Output configuration	249

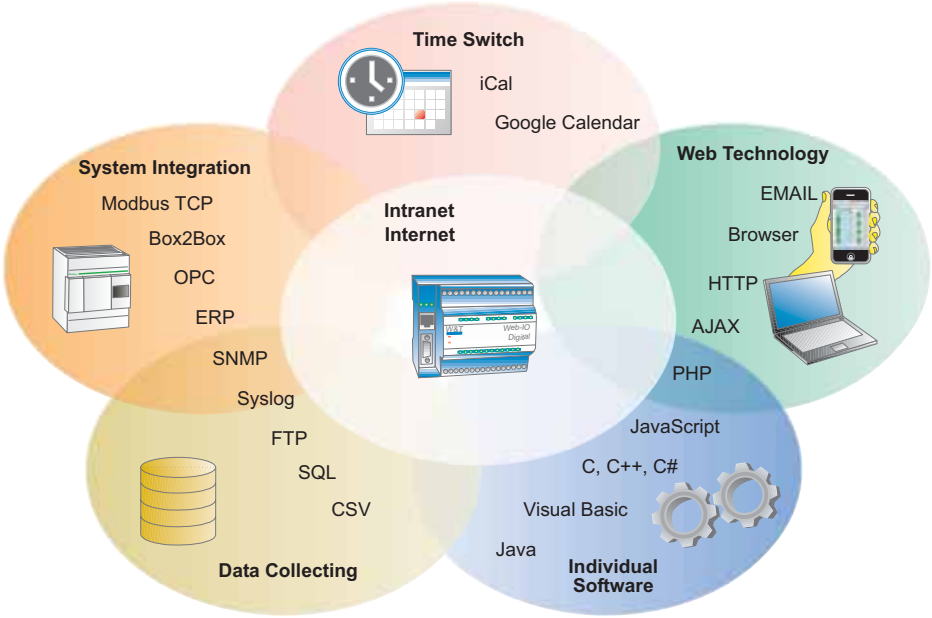
6.3.3 Logic functions 252
6.3.4 Pulse mode 254
6.5 Changing device-specific displays 256
6.6 Upload/Download 258
6.6.1 Download 258
6.6.2 Upload 259
6.7 Diagnostics and testing 261
6.7.1 Error report 261
6.7.2 Configuration 262
6.7.3 LED test 266

7 Appendix 267
7.1 IP address assignment alternatives 268
7.1.1 Assigning the IP address using the ARP command 268
7.1.2 Assigning the IP address through the serial port 269
7.1.3 IP assignment via RARP server 271
7.1.4 Assigning the IP using DHCP protocol 272
7.1.5 Assigning the IP using BOOTP protocol 275
7.2 Firmware Update 277
7.2.1 Where can I get the current firmware? 277
7.2.2 Firmware update over the network using WIndows 277
7.3 Emergency access 279
7.4 Ports and network security 280
7.4.1 Ports 280
7.4.2 The W&T Web-IO and network security 280
7.4.4 Ports with special function: 282
7.5 Timing and signal processing limits 284
7.5.1 Signal detection 284
7.5.2 Signal transmission 284
7.5.3 Signal timing 284
7.6 Network inventory via UDP 286
7.6.1 The info packet 286
7.7 Error messages and causes 289
7.8 Technical Data 291
7.8.1 Technical Data 57630/57631 291
7.8.2 Technical Data 57637, 57638 292
7.8.3 Technical Data 57634 293
7.9 Disposal 294

1. Overview of Potential Uses

Devices in the W&T Web-IO Digital product family provide the user with a number of switching in- and outputs on a screw terminal, with the number depending on the model.

A TCP/IP-Ethernet 10/100BaseT interface gives the Web-IO boxes access to a variety of protocols land applications.



1.1 Overview of potential uses

Depending on the configuration the outputs, inputs and counters on Web-IO boxes can be accessed in various ways.

1.1.1 Browser access

In the simplest case the inputs and outputs on the Web-IO can be accessed directly from the browser. The Web-IO provides finished Web pages for this purpose:

For PC browsers
The home and the user page display the self-refreshing status of inputs, outputs and counters. It is also possible to switch the outputs and clear the counters.

For smart phone
The Smart page provides a very compact overview, optimized for the display size of mobile phones, of the status of inputs, outputs and counters. In addition the outputs can be switched.

Your own Web pages
Of course it is also possible to display the IOs and counters of the Web-IO on your own Web pages and replace the pre-prepared pages with your own.

1.1.2 Box-to-Box

In this mode the outputs on one Web-IO follow the inputs of another. This is how you would send switching signals between two locations over the network.

1.1.3 Time-based switching

The outputs on the Web-IO can be switched on a time basis. There are two ways to configure the switching times:

Use of a local calendar program
Programs such as Outlook or Sunbird allow you to export events (the switching times are treated as events) as an iCal file. The Web-IO can import these iCal files and convert them into switching times.

Scheduling switching times in Google Calendar
The switching times can be specified like normal events in the Google online calendar. The Web-IO then automatically opens the calendar dates, cyclically if desired, from Google and calculates the switching times.

1.1.4 Integration into existing systems

To make it easy to integrate into existing systems, the Web-IO supports the most commonly used communication standards.

OPC
The included product CD provides the free OPC server for W&T devices. This allows Web-IOs to be incorporated into all OPC-capable display and control systems.

Modbus TCP
The Web-IO can if needed work as a Modbus TCP client and provide the status of inputs, outputs and counters using corresponding memory sectors.

SNMP - Simple Network Management Protocol
By supporting SNMP the Web-IO is able to simply pass the switching states to network management systems. It is also possible to send SNMP traps.

An appropriate MIB can be directly downloaded from the Web-IO.

1.1.5 Access from your own applications

The socket interface of the Web-IO offers three possibilities:

Communication via HTTP-Requests
Using HTTP requests the Web-IO can be accessed from Web applications with techniques such as JavaScript, AJAX and PHP. This allows you to create individual applications for the browser.

Communication via command string
In this mode a readable command set based on HTTP is used.

Communication via binary structures
In this mode communication takes place via binary data structures.

In the last two variations the Web-IO can function as a TCP client or server, or even as a UDP peer. All programming languages which permit TCP/IP communication are suitable for programming applications for the Web-IO (C, C++, C#, VB, VB.net, JAVA, Delphi, Lazarus, Pearl, Python, ...).

Even access from embedded systems and PLCs is possible.

1.1.6 Email and more - information when status changes

The Web-IO has an internal alarm system. An alarm can be triggered by the occurrence of certain conditions on the inputs, outputs or counters.

The following methods are available for sending the alarms:

- Email
- SNMP-Trap
- SYSLOG
- Message to TCP-Server
- Message via UDP
- FTP

1.1.7 Acquiring switching states

Messages via the Web-IO alarm system
To capture the switching states of the Web-IO with a time stamp you can also use the alarm system of the Web-IO. In addition to a status change in the IOs, a timer can be used as the trigger. Practical methods of data storage are:

- SYSLOG
- Message to TCP-Server
- Message via UDP
- FTP

E-mail and SNMP traps can also be used.

Acquiring IO states in the W&T Motherbox
The W&T Motherbox is a Web server which also provides a MyS-QL database for storing the data from Web-IO products.

There are two recording models to choose between:

- The Motherbox cyclically polls the associated Web-IO and stores the obtained data in a fixed time grid.
- The Web-IOs send their data to the Motherbox when there is a status change.

1.1.8 Logic functions - linking inputs and outputs

Another way to switch the outputs is to use logical operations between inputs and outputs. The status of an output can be switched to ON or OFF depending on a defined input pattern.

1.1.9 Basic port behavior

Inputs
The inputs are configured for 24V automation signals but can be switched using voltages between -30V and +30 V. An ON signal is reliably detected from 8V +/-1V.

A configurable time filter can be used to eliminate the effects of contact bounce and noise pulses.

Outputs
Models 57630M, 57631M, 57637 and 57636 have current sourcing outputs with a load capacity of 500mA. To drive larger currents, up to four outputs can be combined into a common switching group. Model 57634M switches the outputs with max. 5A via relays.

The outputs can be configured so that they resume the quiescent state after switching following a specified time (pulse mode)

1.1.10 General basic functions

In addition to the functions of the Web-IO which directly access inputs and outputs, there are several important system functions. These include:

- Web-Based Management
- System clock incl. time server synchronization
- Firmware updates

All functions introduced here in brief are described in greater detail in the following sections.

2 Before starting up

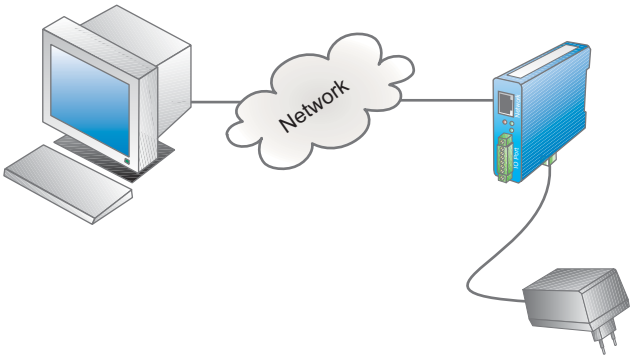
We recommend to read the complete Manual for the Digital Web-IO before startup.

- Connecting the supply voltage
- Connecting to the network
- Assigning the IP address

2.1 Supply voltage and network connection

2.1.1 #57637 and #57638

Use a patch cable to connect the Web-IO to an unused HUB or SWITCH port. The Web-IO has an autosensing 10/100BaseT network connection and automatically detects the network speed.



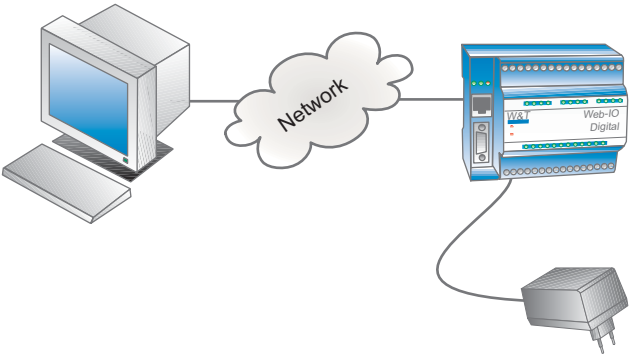
Connect a supply voltage between 12 V and 24 V AC or DC to the power terminal on the underside of the Web-IO.

Compatible power supplies can be obtained as an accessory from W&T. See *http://www.wut.de Products & Downloads >> Web-IO Digital.*

After approx. 2 sec. a beep will sound and the Status and Error LEDs should have turned off. If the red Error LED flashes, check the network connection.

2.1.2 #57630, #57631, #57634 and #57636

Use a patch cable to connect the Web-IO to an unused HUB or SWITCH port. The Web-IO has an autosensing 10/100BaseT network connection and automatically detects the network speed.



Connect a supply voltage of between 12 V and 24 V AC or DC to terminals 1 and 2.

Compatible power supplies can be obtained as an accessory from W&T. See *http://www.wut.de*

After approx. 2 sec. a beep will sound and the Status and Error LEDs should have turned off. If the red Error LED flashes, check the network connection.

If the WEB-IO has no IP address or address 0.0.0.0, the on error *and* system error *LEDs remain on after a reset or new start! The LEDs will turn off only if an IP address has been assigned.*

2.2 Assigning the IP address

The Web-IO is factory set to IP address 0.0.0.0. Before assigning a new one, you must obtain an appropriate IP address from your system administrator. In smaller, unrouted networks use the IP address of your PC and simply change the last digit.

2.2.1 Automatically obtaining the IP address via DHCP

By factory default setting the Web-IO automatically attempts to obtain an IP address from a DHCP server upon system startup. The following parameters are sent:

- Subnet mask
- Gateway address
- DNS-Server
- Lease-Time

Please note that the Web-IO may be assigned a different IP address upon restart. To be able to specifically contact the Web-IO, an address reservation or a fixed association between device name and assigned IP address should be made in the DHCP server.

Additional details about DHCP can be found in the section *Alternative IP address assignment.*

2.2.2 Assigning the address using the Wutility-Tool

This method of address assigning can only be used on a Windows-based PC. If no Windows system is available, note the section Alternative IP address assigning *in the Appendix*

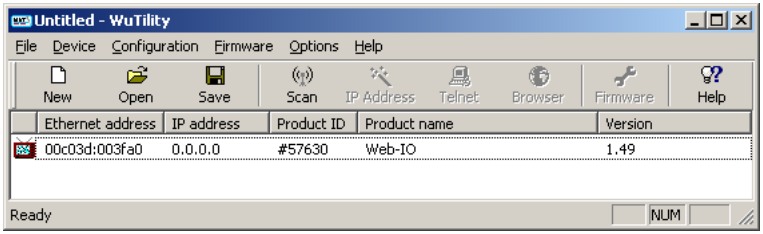
The WuTility installation files are found on the product CD, whereby you enter the article number of your Web-IO into the *Search* field after starting the CD.

Alternately you can download WuTility from www.WuT.de (using Article Search *and* Tools*).*

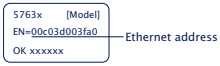
Start WuTility by double-clicking oin the *Easy Start* icon on the desktop.

Wutility now automatically scans your network for available W&T devices.

The list of found W&T network components should now contain an entry for the Web-IO. The IP address should be indicated as 0.0.0.0 if you are starting up for the first time. Use the displayed Ethernet address to ensure that the entry corresponds to your Web-IO.

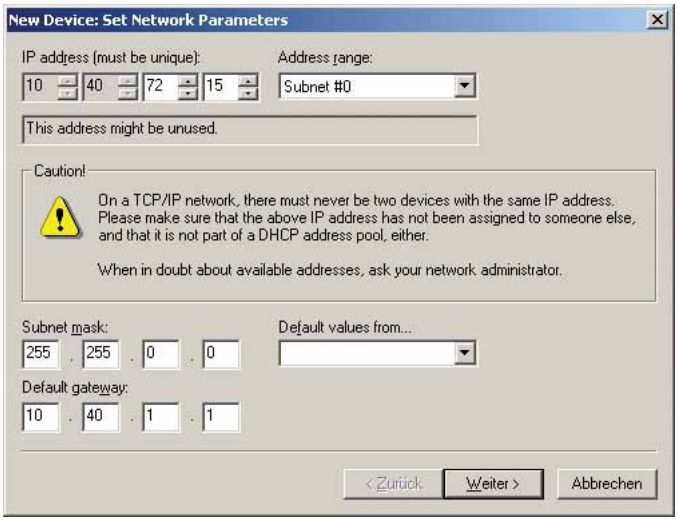

Untitled - WuTility
File Device Configuration Firmware Options Help
New Open Save Scan IP Address Telnet Browser Firmware Help
Ethernet address IP address Product ID Product name Version
00c03d:003fa0 0.0.0.0 #57630 Web-IO 1.49
Ready
NUM

Read the Ethernet address of the Web-IO from the label on the side of the housing:



Highlight the entry for your Web-IO.

Click on the IP address icon and enter the IP address you want to assign to your Web-IO.



Check whether the entries for subnet mask and gateway are correct and confirm with *Continue*. Once all the settings have been correctly added, the Web-IO appears with the desired IP address in the list.

After scanning the network again Wutility should update the display in full.

All other settings can now be made conveniently from the browser using Web-Based Management.

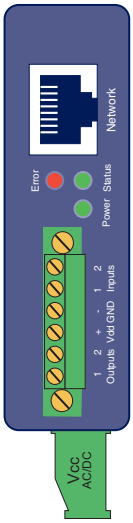
After loggin in as Administrator, use the mode *Profiles* to configure standard application cases. Detailed descriptions for configuring the various modes can be found in the following sections.

3 Connections, operating elements and LEDs

- Supply voltage connection
- Input and Output connections
- Ethernet connection
- RS232
- Device status LEDs

3.1 Connecting Web-IO 2xDig. In, 2xDig. Out PoE

Type 57637, 57638



3.1.1 Supply voltage input (Vcc)

The supply voltage is coinected to the screw terminals on the underside of the housing. The AC adapters availble from W&T are equipped on the low-voltage side with a special plug.

Alternately an availble voltage of between 12V and 24V for power can be connected. Since the Web-IO can be operated on AC or DC voltage, polarity of the supply voltage is non-critical.

To prevent cable break, we recommend fitting the wire ends with crimp contacts.

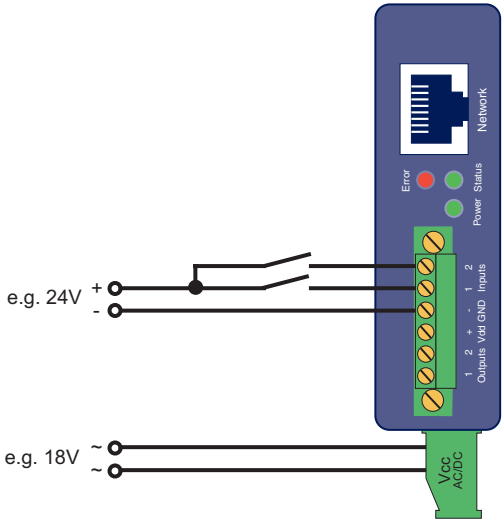
The terminals Vdd on the front side power only the switching outputs and require a DC voltage. Please refer to the section „Output wiring“.

3.1.2 Input wiring

The Web-IO 12xDigital is equipped with 2 digital inputs.

The permissible input voltage range is +/-30V referenced to ground.

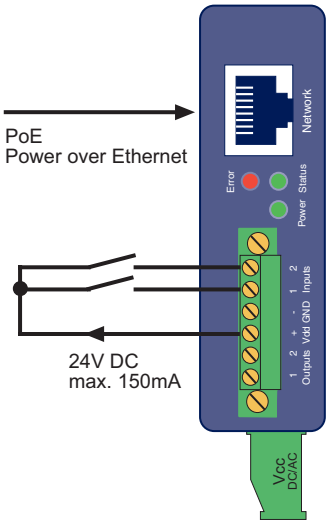
The switching threshold for the inputs is 8V +/- 1V. Voltages less than that are recognized as OFF (0 signal). Voltages above 8V are procesed by the Web-IO as ON (1 signal). Input voltages between 7V and 9V should be avoided, since they represent no defined state.



The connection example shows how two inputs are driven by signals from different circuits without PoE. Here it is important that the input signals have the same reference to ground.

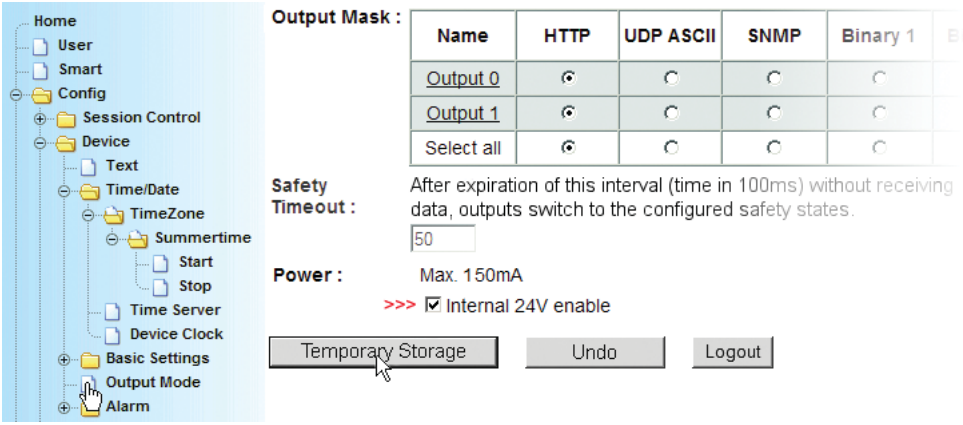
Alternately it is possible to power the device using PoE. In this case the internal auxiliary supply can be used for example to switch a signal to the inputs through potential-free contacts.

The Web-IO 2x Dig. Inp, 2x Dig. Out PoE 24V DC provides 24V DC on terminals 3 and 4 of the front terminal strip. These have a maximum load capacity of 150mA.



The auxiliary voltage must first be enabled using Web Based Management.

Log in as administrator and select *Config >> Device >> Output Mode* from the menu tree.



Select *Power > Internal 24V enable*

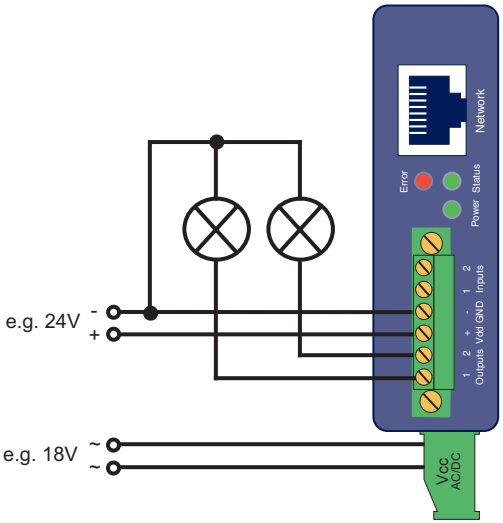
You must then save the changed setting in the Web-IO using *Logout* and *Save*.

The input lines should not be longer than necessary. Unshielded cable may be used for cable lengths less than 2m.

For longer cable runs you should always take suitable shielding measures against noise interference. We recommend the use of shielded cable and/or ferrite rings as well as separate cable routing to ensure that noise signals are not interpreted as desired signals.

In addition to detecting the input status (ON/OFF), each input is also equipped with a counter. In the factory default configuration edges (state change) are counted. The counter may alternately be reconfigured from edge to pulse counting. In addition, 2 inputs can be used together for incremental, directional counting. Here the counter which first detects the change is incremented.

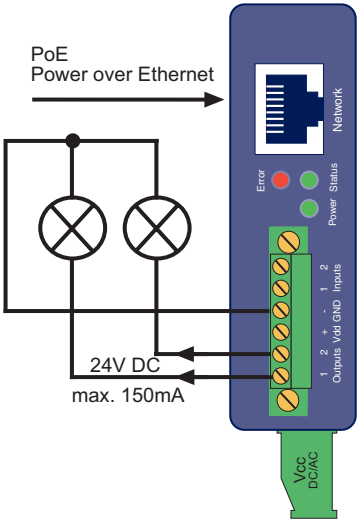
3.1.3 Output circuit



The 2 outputs on the Web-IO are current sourcing. The supply voltage for the outputs may be between +6 V and +30 V (DC) and is introduced on terminals Vdd and GND.

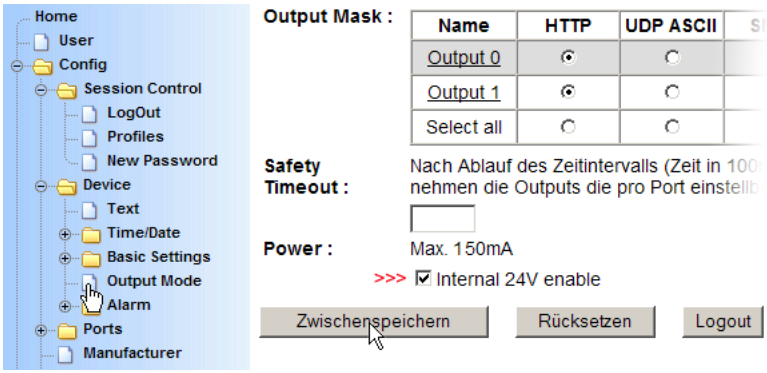
Each output should switch a maximum of 500mA. To achieve higher switching currents, both outputs can be combined and switched together. The maximum switching current is the sum of the individual currents. For 2 outputs this is max. 1A.

When powering using PoE the outputs may alternately be supplied by the internal 24V auxiliary power.



In this case both outputs together have a maximum switching current capacity of 150mA. The auxiliary power is protected by a self-resetting thermal fuse. In case of overload the voltage is turned off and is not available again until several minutes after removing the load.

The internal auxiliary power must first be activated using Web Based Management. Log in as administrator and select *Config >> Device >> Output Mode* in the menu tree.



Select *Power > Internal 24V enable*

You must then save the changed setting in the Web-IO using *Logout* and *Save*.

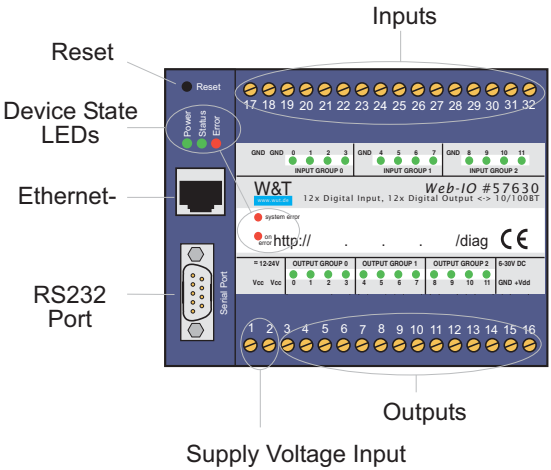
Outputs which are switched with an inductive load such as a relay should be protected with a free-running diode.

The outputs feature thermal overload protection and are short circuit protected. In the OFF state there is also a test to see whether a load is connected to the outputs. If a load is connected, a leakage current of approx. 10µA flows. This may result in slight illumination of the LEDs even in the quiescent state.

Web-based management can be used to activate load monitoring which tells the user when there is no load (e.g. cable break) or overload.

When dimensioning the output voltage supply you should take the required current into account. If the selected output voltage varies between 12V and 24V, Vcc and Vdd can be provided from the same source.

3.2 Connecting Web-IO 12xDig. In, 12xDig. Out



3.2.1 Supply voltage input

The supply voltage is connected to screw terminals 1 and 2. The included 230V plug-in power supply is equipped with a special plug on the low-voltage side.

Alternately an existing voltage of between 12V and 24V can be connected. Since the Web-IO can operate on AC or DC voltage, the polarity of the supply voltage is not an issue.

To prevent cable breaks, we recommend fitting the ends of the conductors with crimp contacts.

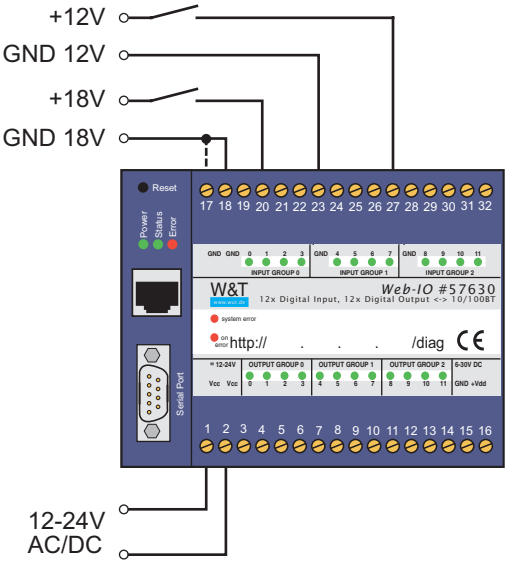
Terminals 15 and 16 supply only the switched outputs and require a DC voltage. Please read the section Input and Output connections and LEDs *before applying power 3.6 Input- and Output connections*

3.2.2 Input circuit

The Web-IO 12xDigital has 12 digital inputs divided into 3 groups of 4 inputs. The groups are galvanically isolated from each other for up to 2kV. A reference ground is brought out for each input group.

The permitted input voltage range is +/-30V with respect to the corresponding reference ground.

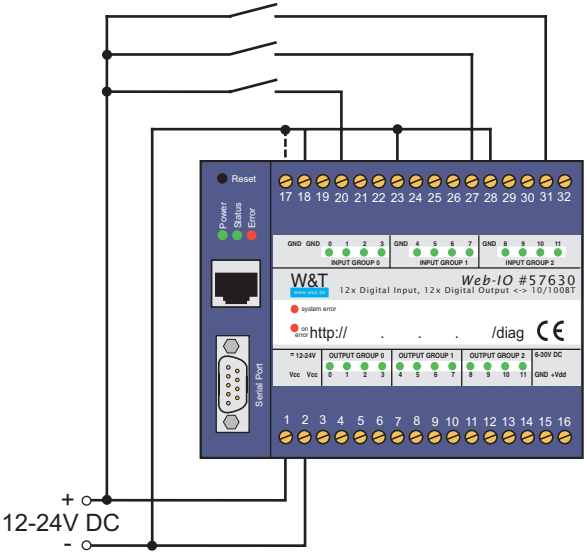
The switching threshold of the inputs is 8V +/- 1V. Voltages less than this are interpreted as an OFF or 0 signal. Voltages over 8V are interpreted by the Web-IO as ON or 1. Input voltages between 7V and 9V should be avoided, since they cannot be definitively interpreted.



Example of an input circuit

This connection example shows two inputs being driven with signals from different circuits. It is important that all signals for an input group have the same ground reference. For signals with with a different reference ground, the circuit should be configured in separate input groups as shown in the example.

If the inputs will be monitoring the states of potential-free contacts, the supply voltage Vcc can be connected to the inputs through the contacts. In this case VCC must be connected to a DC voltage of 12-24V.



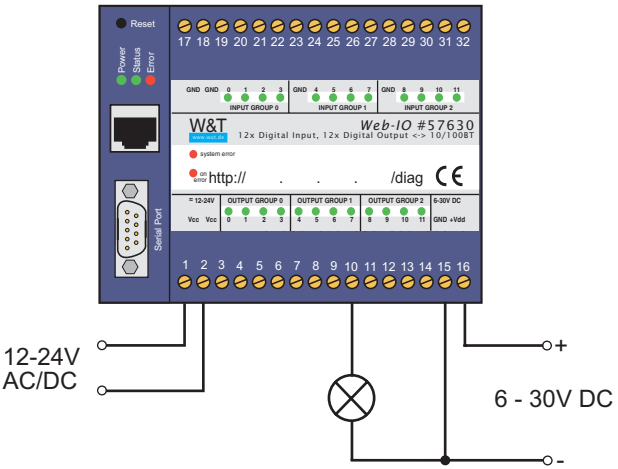
The cables should not be longer than necessary. Unshielded cables can be used for lengths under 2m.

Longer runs should always use appropriate shielding measures to protect against noise. We recommend using shielded cable and/or ferrite cores as well as separate cable runs to prevent noise signals from being interpreted as actual signals.

In addition to detecting the input status (ON/OFF), each input also has a counter. As shipped the edges are counted (state change). The counters in the first input group can alternately be changed from edge to pulse counting.

3.3.3 Output circuit

The 12 outputs of the Web-IO are current sourcing. The supply voltage for the outputs can be between +6 V and +30 V and is brought in through terminals 16 (Vdd) and 15 (GND).



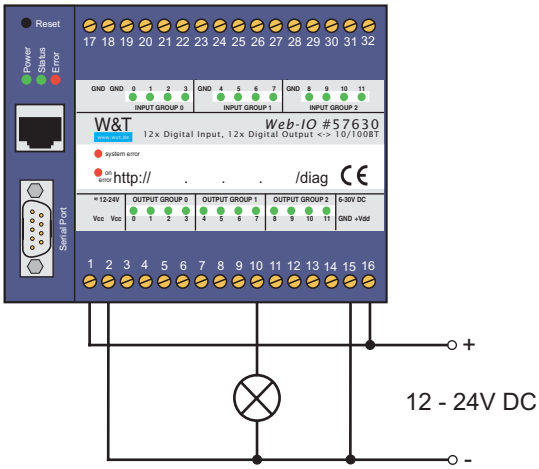
Each output can switch a maximum of 500mA. To achieve higher currents, individual outputs can be combined into groups of 2 or 4. The maximum switching current is the total of the individual currents: max. 1A for 2 outputs, max. 2A for 4 outputs

The outputs have thermal overload protection and are short-circuit protected. In the OFF state there is also a check to see whether a load is present on the outputs. When a load is connected, a current of approx. 10µA flows even when the output is turned off. This means that the LEDs associated with the outputs may glow slightly even when the output is OFF.

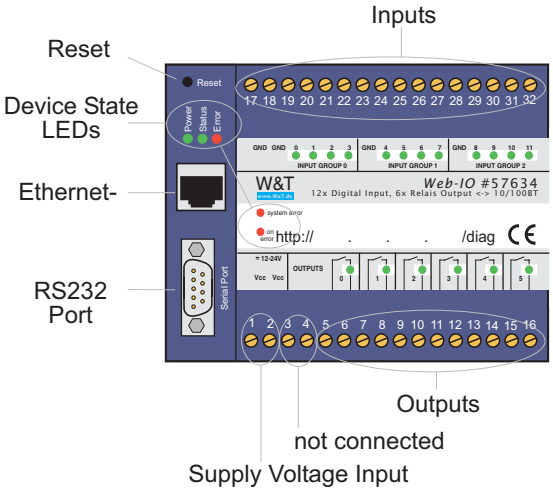
Web-based Management can be used to activate load monitoring, which lets the user know when there is no load (e.g., cable break) or overload.

Outputs which are switched with an inductive load such as a relay should be protected with a free-running diode.

When dimensioning the output voltage supply, the required current should be taken into consideration. If the selected output voltage varies between 12V and 24V, Vcc and Vdd can also be supplied from the same source.



3.4 Connecting Web-IO 12xDig. In, 6xDig. Relais



3.4.1 Supply voltage input

The supply voltage is connected to screw terminals 1 and 2. The included 230V plug-in power supply is equipped with a special plug on the low-voltage side.

Alternately an existing voltage of between 12V and 24V can be connected. Since the Web-IO can operate on AC or DC voltage, the polarity of the supply voltage is not an issue.

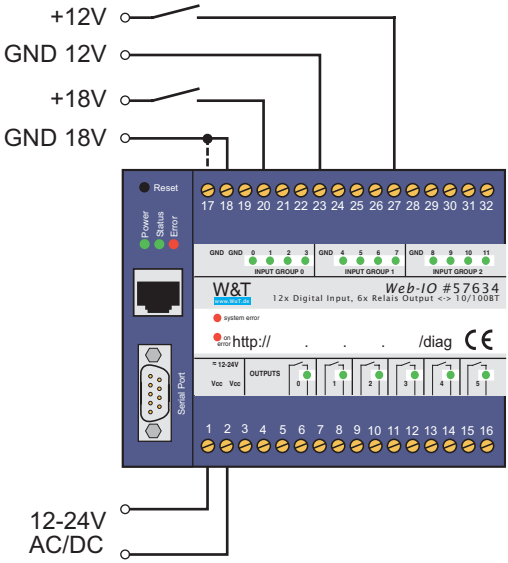
To prevent cable breaks, we recommend fitting the ends of the conductors with crimp contacts.

3.4.2 Input circuit

The Web-IO 12xDigital has 12 digital inputs divided into 3 groups of 4 inputs. The groups are galvanically isolated from each other for up to 2kV. A reference ground is brought out for each input group.

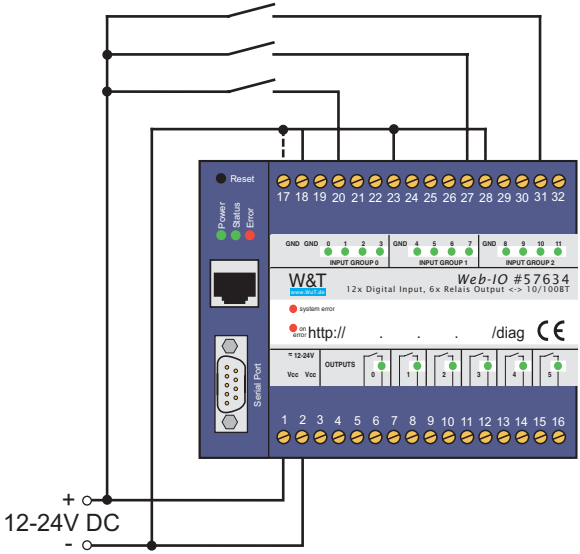
The permitted input voltage range is +/-30V with respect to the corresponding reference ground.

The switching threshold of the inputs is 8V +/- 1V. Voltages less than this are interpreted as an OFF or 0 signal. Voltages over 8V are interpreted by the Web-IO as ON or 1. Input voltages between 7V and 9V should be avoided, since they cannot be definitively interpreted.



This connection example shows two inputs being driven with signals from different circuits. It is important that all signals for an input group have the same ground reference. For signals with with a different reference ground, the circuit should be configured in separate input groups as shown in the example.

If the inputs will be monitoring the states of potential-free contacts, the supply voltage Vcc can be connected to the inputs through the contacts. In this case VCC must be connected to a DC voltage of 12-24V.


Reset
Power
Status
Error
17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32
GND GND 0 1 2 3 GND 4 5 6 7 GND 8 9 10 11
INPUT GROUP 0
INPUT GROUP 1
INPUT GROUP 2
W&T
Web-IO #57634
12x Digital Input, 6x Relais Output <-> 10/100BT
system error
on error
http:// . . . /diag
~ 12-24V
Vcc Vcc
OUTPUTS
0 1 2 3 4 5
Serial Port
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
+
12-24V DC
-

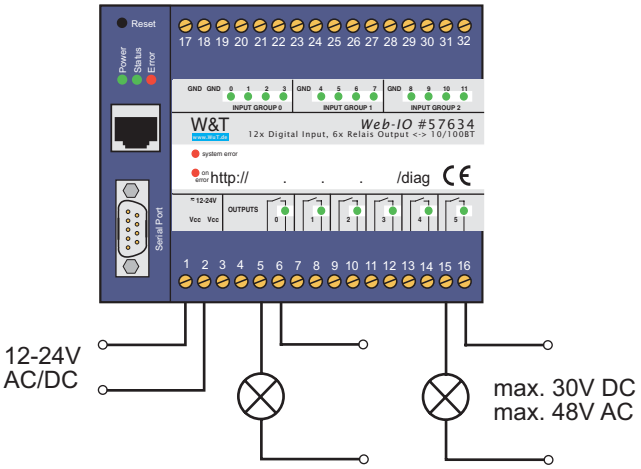
The cables should not be longer than necessary. Unshielded cables can be used for lengths under 2m.

Longer runs should always use appropriate shielding measures to protect against noise. We recommend using shielded cable and/or ferrite cores as well as separate cable runs to prevent noise signals from being interpreted as actual signals.

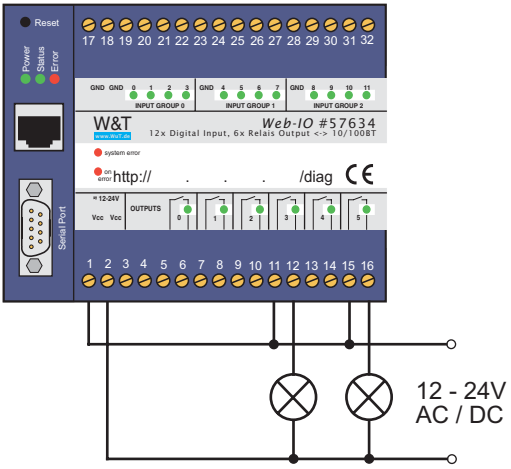
In addition to detecting the input status (ON/OFF), each input also has a counter. As shipped the edges are counted (state change). The counters in the first input group can alternately be changed from edge to pulse counting.

3.4.3 Output circuit

The 6 outputs of the Web-IO are relay contacts (N.O.). The switch voltage for the outputs can be up to 30V DC (150W) or 48V AC (240W).

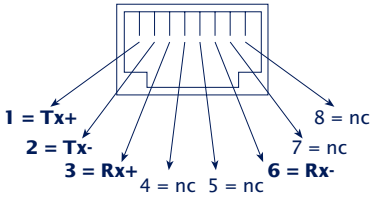


When dimensioning the output voltage supply, the required current should be taken into consideration. If the selected output voltage varies between 12V and 24V, Vcc and Vdd can also be supplied from the same source.



3.6 Ethernet connection

The Web-IO models provide an IEEE 802.3-compatible network connection via a shielded RJ45 connector. The pinouts correspond to an MDI interface, so that the connection to the hub or switch can be made with a 1:1 shielded patch cable.



RJ45-Buchse (Belegung AT&T256)

10/100BaseT, Autosensing

Both 10BaseT and 100BaseTx standard with a bit rate of 100MBit/s and the possibility of full-duplex transmission are supported. Changing between the two network speeds is automatically done by the autosensing function of the Web-IO depending on the possibilities of the hub or switch used. The prerequisite for operation at 100MBit/s is suitable cabling (at least conforming to Cat. 5/ISO Class D).

Both stations (Switch and Web-IO) should be run in auto-negotiating mode.

Galvanic isolation

The network connection is galvanically isolated with respect to the supply voltage and to the digital IOs and the serial port(s) up to at least $500V_{rms}$.

Link status

The Error LED on the front of the unit indicates the current link status: If it flashes at 1-2 second intervals, there is no connection to the hub or the connection is faulty.

Manageable switches often have special protocols (Spanning Tree Protocol, Port-Trunking, ...), as required for example for uplinks to other switches or for broadband connection of servers. These protocols are not required for

connecting normal terminal devices such as the Web-IO, but do under some circumstances delay opening of communication considerably after a restart.

In some cases it can even happen that switches do not pass data which the Web-IO sends after the physical links are established (e.g. ARP requests and BootP packets)!

We recommend deactivating the unneeded protocols and functions of the switch on the port used for the Web-IO. Please consult the responsible network administrator.

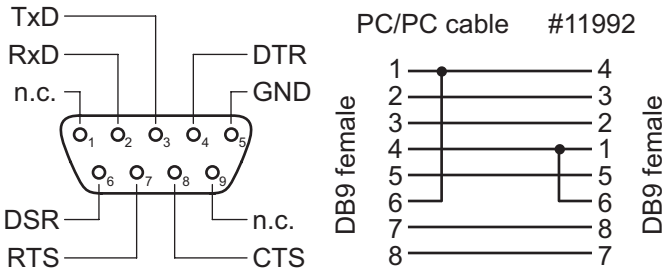
3.7 Serial connection

In the Web-IO 12xDigital without Com-Server functionality and in the Web-IO 24xDigital OEM the serial port is available only for configuration purposes such as IP address assignment.

The pinouts of this RS232 port are identical to those of your PC. This lets you use standard cables.

A detailed pin description as well as the W&T cables required for most common applications can be found in the illustrations below.

RS232 pin assignments and functions, DB9 plug



To serially assign the IP address, cable 1199x and 1179x are recommended.

The Web-IO 2xDigital does not have a serial port.

3.8 LEDs for device status

Power-LED:
Indicates the presence of supply voltage. If the LED does not come on, check the power supply connections.

Status-LED:
Blinks whenever there is network activity with the Web-IO. Periodic blinking indicates that the port has a connection to another station.

Error-LED:
The Error-LED uses various blinking codes to indicate error states on the device or network port.

1x blinking of the Error-LED
Check network connection. The Web-IO is not getting a link pulse from a hub/switch. Check the cable or the hub/switch port.

2x or 3x blinking of the Error-LED
Press the button on the housing to perform a reset (on the Web-IO 2xDigital by interrupting power). If this does not clear the error, reset the device to its factory defaults. Since this resets all network settings, you should write down your settings first.

Web-Based Management: *Config >> Session Control >> LogOut >> Restore Defaults* . Following a reset the device is restored to its factory defaults. Reconfigure your network settings.

More information about menus and coniguration can be found in the section Web-Based Management.

on error http://xxx.xxx.xxx.xxx/diag-LED:
Indicates internal configuration errors. For troubleshooting, go to page http://<IP address of the Web-IO>/diag in the unit.

For example:

```
http://172.16.232.17/diag
```

system error:
Serious hardware error. Try to restart the unit by disconnecting the power supply. If the condition persists, please return the unit to your dealer so that it can be checked by W&T.

Power-LED +Status-LED +Error-LED on
= Self-test error
The self-test performed by the Web-IO after each start or reset could not be correctly finished, due for example to an incomplete update of the firmware. In this condition the device is no longer functional. Please return the unit to your dealer so that it can be checked out by W&T.

If the Web-IO has no IP address or the address is 0.0.0.0, the LEDs on error *and* system error *will remain on after a reset or new start! The LEDs will turn off only if an IP address is assigned.*

The Web-IO 2xDigital does not have the on error *and* system error *LEDs*

4 Configuring the network parameters

- Web-Based Management
- Basic setting of the network parameters

4.1 Web-Based Management

The W&T Web-IO Digital provides the user with a complete Web-Based Management system. This means: If the Web-IO has an IP address, all the settings can be conveniently made over the network from the browser.

4.1.1 Structure of the Web site

When the Web-IO is opened in the browser, a 2-part Web site always appears. These two parts are also known as frames.

On the left side is always the navigation frame, which one could compare with the table of contents of a book.

To the right is the display and configuration frame. When the web site is first opened, this contains the current status of the inputs and outputs of the Web-IO.

4.1.2 Navigation

The navigation frame contains a directory tree in which all the available menu items of the Web-IO are listed by categories. Clicking with the mouse either displays additional menu items for a category and/or a new content is built in the configuration frame.

Avoid using the „Forward“ and „Back“ buttons of your browser. Use only the navigation tree or buttons and links in the configuration frame to navigate. Otherwise you could lose parameters buffered in the background when you jump forward or back.

4.1.3 Easier navigation with profiles

The Web-IO 12xDigital offers a variety of operating mode and configuration possibilities. And to get even the first-time user

of the Web-IO off to a good start, we have created profiles for the most common applications.

After successfully loggin in as an Administrator (the next section will cover the possible access privileges), the user is given a choice between configuration using profiles and expert mode, which does not provide any configuration assistance..

Working with profiles simplifies configuration by color highlighting the configuration points which are actually needed for the corresponding mode. Both the menu entries in the navigation tree and the respective parameters on the configuration pages are highlighted.

4.1.4 Access rights

Depending on the login, the Web-IO distinguishes among 3 different access rights:

Default User:
Anyone opening the start page of the Web-IO has this status initially. The status of the inputs and outputs can be displayed but not modified. No changes to the configuration are possible.

Administrator:
Clicking on *Config* in the menu tree and entering the administrator password gives you full access to the Web-IO. The start page is refreshed with a significantly expanded menu tree. As shipped it is not necessary to enter a password, it is sufficient to click on the Login button.

The next sub-section describes assigning passwords in greater detail.

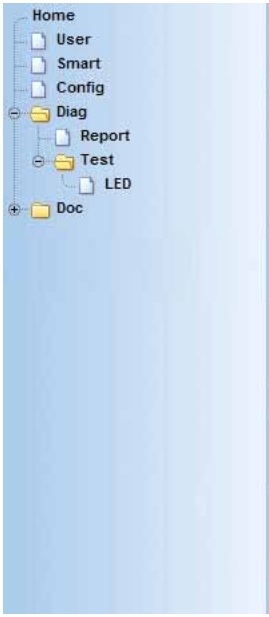
Operator:
The access rights of the operator are limited to setting the outputs, clearing the counter, changing the alarm outputs and a few diagnostic activities. The operator has no additional configuration options. The login in this mode is also

accomplished by entering the password selected for this access.

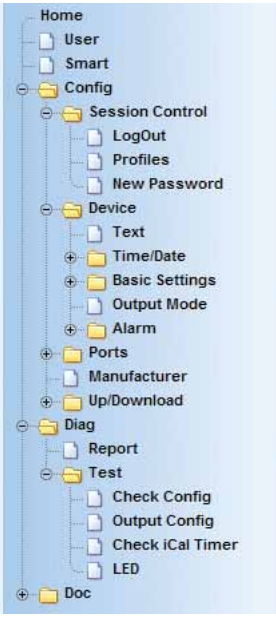
The login is done after opening the Web-IO using Config in the menu tree.

Here again a the menu trees for the various access rights by way of comparison:

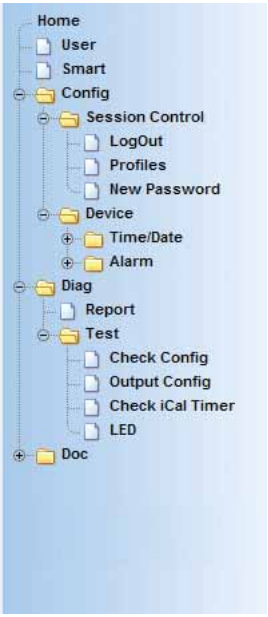
Default User



Administrator



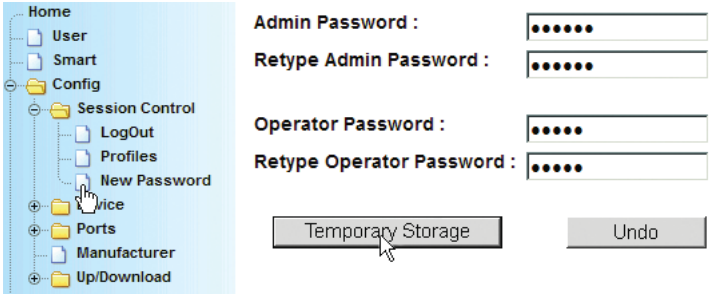
Operator



4.1.5 Changing and creating the access passwords

The user must be logged in as an Administrator.

In the navigation tree select: *Config >> Session Control >> New Password* and enter the desired passwords in the fields. A password is limited to 31 characters.



It is not permitted to use the same password for both Admin and operator access.

Clilcking on the *Temporary Storage* button sends the new passwords to the Web-IO. You still have to activate the changes.

Tip: Keep a copy of the passwords in your personal documents.

If you have forgotten the passwords, see the Appendix of this manual for help.

4.1.6 Saving and activating settings

To increase you understanding of the sequences in Web-Based Management, first a few basic remarks concerning HTTP protocol:

With HTTP the browser opens a TCP connection to the HTTP server, requests Web pages and contents from the server or sends data to the server. The server then immediately closes the connection.

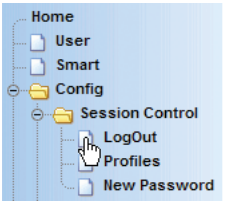
Entries made by the user on a Web page are first temporarily stored by the browser and only sent to the server after clicking on the *Submit* or *Temporary Storage* button through a new TCP connection.

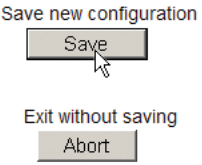
If for example the browser is closed before the *Temporary Storage* button is clicked, the entries are irretrievably lost. This can

also happen if the Web site is changed without sending the entries.

Nearly all the Web sites that can be opened from the navigation tree offer the possibility of making or changiong entries. Once you are finished with the entries on such a configuration page, you must first click on the *Temporary Storage* button and only then change to a new Web site.

Once all the entries and configuration changes have been made, the new settings must still be activated. Use the *Logout* button located on each configuration page, or use the menu path Config >> Session Control >> LogOut.





After clicking on the Save button, all the settings are updated in the Web-IO and the start page is refreshed in Default User mode. To make additional setting changes, you must log in again with the necessary access rights.

Login Timeout

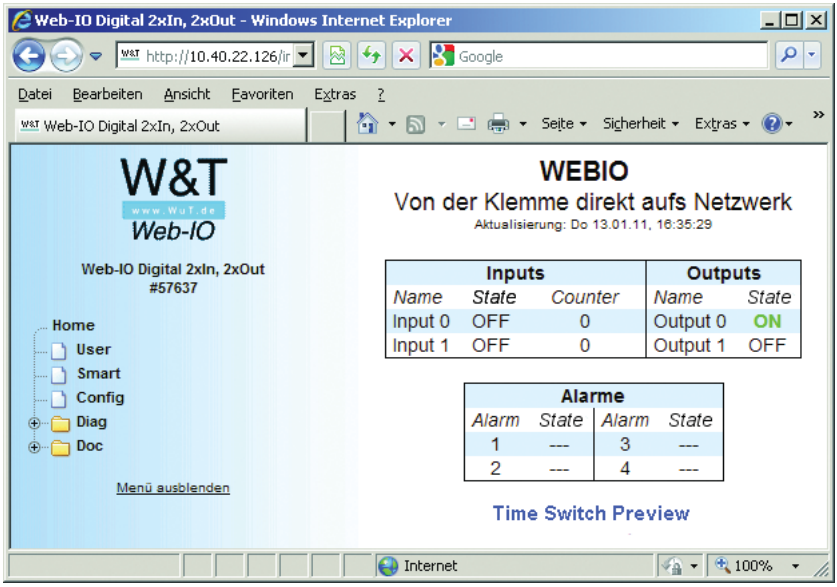
If no entry is made in Adminstrator or Operatotmode for more than 60 minutes, the Web-IO resets itself to Default User rights. All entries and changes made to this point, even if they were already sent to the Web-IO, are lost.

4.2 Basic setting of the network parameters

Whenever a Web-IO will be used beyond the borders of the local network, a few additional network parameters are needed in addition to the IP address.

To be able to perform the configurations described at this point, the Web-IO should already have an IP address for the network. If this is not the case, or if the current IP address of the Web-IO is unknown, first perform the steps described in section 2.

If the Web-IO already has an IP address, enter this in the URL field of your browser. After a few seconds the start page of the Web-IO will appear.



Select *Config* in the navigation menu.

You are now prompted to enter the correct administrator password. As shipped there is no password required, and it is sufficient to click the *Login* button.

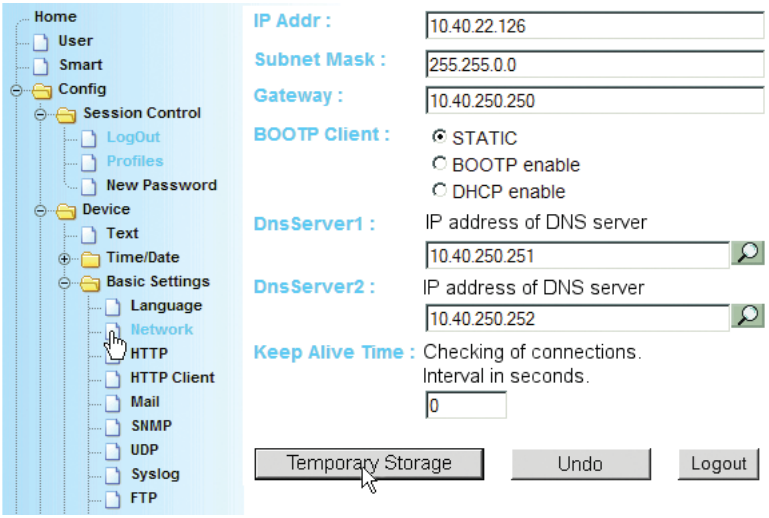
Once the correct password has been entered, the Web-IO confirms successful login. At this point you can choose between Expert mode and working with profiles. Use the *Profiles* button.

In the Web page which now appears select the radio button *Basic network parameters* and confirm by clicking on the *Show profiles* button.

In the navigation menu now select the color highlighted item *Config >> Device >> Basic Settings >> Network*.

Now go to the navigation menu and select *Config >> Device >> Basic Settings >> Network*.

The following screen appears:



Enter here the required network parameters.

IP Addr.
The IP address may be changed at this point.

Subnet Mask / Gateway
For data exchange throughout the subnet, the appropriate subnet mask and IP address of the gateway must be entered.

DHCP Client / BOOTP Client
The factory default configuration is for the Web-IO to be able to obtain its IP address and other necessary network parameters from a DHCP server.

In this case the Web-IO receives any IP address from the address pool of the DHCP server unless a special address reservation has been made. Assured use of the Web-IO server services is then only possible if there is automatic synchronization in the network between the DHCP and DNS server. The Web-IO is then uniquely addressable using the Device Name.

Alternately, the Web-IO can obtain an IP address using the older BootP protocol, but this requires an address reservation on the DHCP/BootP server for trouble-free operation.

If the Web-IO will not be taking part in central IP address assignment via DHCP or BootP, choose static.

DNS Server
With some network partners, for example mail and time servers, it is recommended that these not be addressed using their IP address but rather using a name. A DNS server is required to resolve names into IP addresses.

If DHCP/BootP is used for assigning the network parameters, the Web-IO also gets the address of the DNS server automatically assigned.

The field DNS Server by DHCP is filled in by the Web-IO. The user can not make changes to this entry.

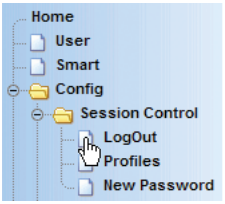
There are two additional fields available for manual entry of DNS servers.

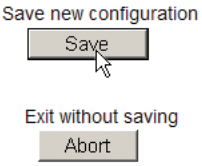
To search an IP address for a name, the Web-IO first turns to the automatically assigned DNS server. If the latter cannot be reached or if the entry is missing, the Web-IO tries DNS servers 1 and 2.

Using the Button you can ping the entered server. This allows you to verify whether the entered server is accessible by the Web-IO. If the entered server is not located in the same subnet, the condition is that subnet mask and gateway have first been stored in the Web-IO using Logout >> Save.

If you do not have the required information, contact your network administrator.

When you have made all your entries, click on the *Temporary Storage* button. This sends the parameters first to the Web-IO. The changes do not become effective until they have been activated. For this use the *Logout* button or menu path Config >> Session Control >> LogOut.





After clicking on the *Save* button, the Web-IO is restarted with the current parameters. Normally the start page of the Web-IO is refreshed within about 1 0 seconds. If it does not refresh, click on the *here* link.

Note that applications which are communicating with the Web-IO at this point in time may experience a fault.

The Web-IO is now set up to the point that it can also be accessed throughout the subnet.

Changing the Device Name

By default each Web-IO has an individual device name which consists of the designation *WEBIO-* followed by the last 3 places of the Ethernet address.

For example, the factory set device name of a Web-IO having Ethernet address 00:c0:3d:01:02:03 is *WEBIO-010203*.

The device name is required especially in networks in which the Web-IO obtains its IP address from a DHCP server which does automatic synchronizing with the DNS system.

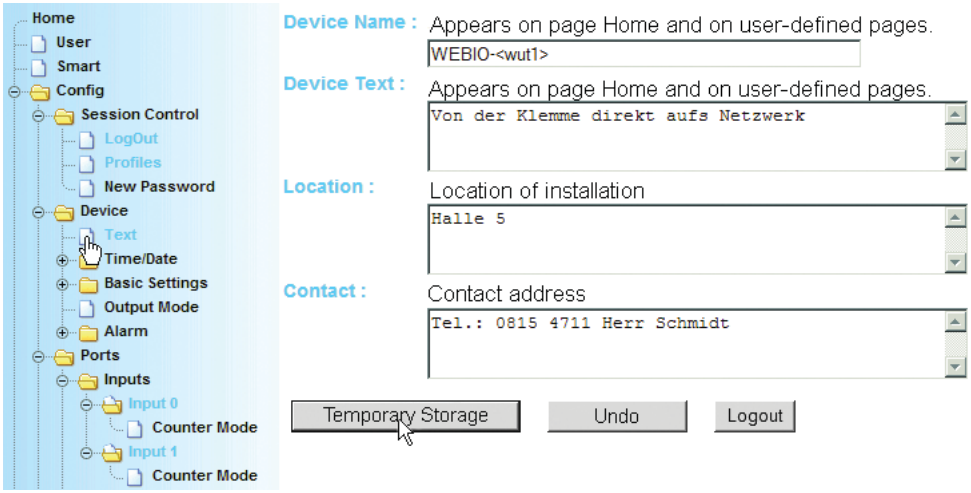
The Web-IO identifies itself within the DHCP protocol using its device name.

Using Web-Based-Management the device name can be changed at any time.

Access rights: *Administrator*

In the navigation menu select *Config >> Device >> Text.*

The following input screen appears:



Device Name
System Name with which the Web-IO reports to a DHCP server using DHCP protocol and which is also displayed on the homepage.

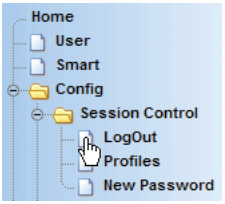
In networks with DHCP and DNS the name entered here should be unique!

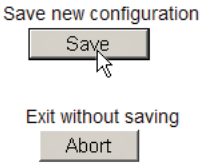
Device Text
The administrator can place a text here which describes the Web-IO and its function. This text is displayed on the homepage of the Web-IO.

Location
Here you can enter the installed location of the Web-IO to make it easier to locate and maintain it in larger installations.

Contact
Here you can enter for example an e-mail address or telephone number at which the responsible technician can be reached when service is needed.

To save the texts, click on the *Apply* button and in the Config menu select >> Session Control >> LogOut.





After clicking on the *Save* button the Web-IO is restarted using the current parameters.

If you like to learn more about the expressions and names used, ask for a free copy of our book TCP/IP-Ethernet and Web-IO, where the necessary basics for starting up network components are explained. WWW.WUT.DE

5 Basic Functions

- HTTP - Controlling inputs and outputs in the browser
- Box-to-Box
- Time-based switching
- Integration into existing systems
- E-mail and more - information when status changes - alarms
- Access from your own application

5.1 HTTP - Controlling IOs in the browser

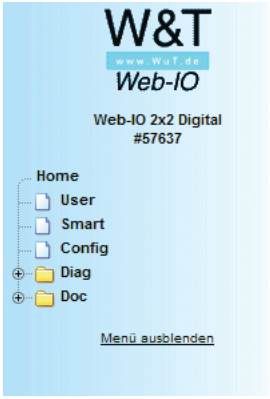
Access from the browser is the simplest way for the user to monitor inputs and counters and to control the outputs.

The Web-IO offers 3 standard views and Web pages:

- Home - For viewing the IO states and alarm status
- User - For viewing the IO states and as needed the available Modbus memory
- Smart - IO status optimized for smart phone display

5.1.1 The Home page

The Home page provides the user with the status of the inputs, counters, outputs and configured alarms, all self-refreshing.



WEBIO-05372A

Von der Klemme direkt aufs Netzwerk

Aktualisierung: Fr 28.01.11, 10:34:13 *

Inputs			Outputs	
Name	*Status*	*Counter*	*Name*	*Status*
Input 0	OFF	0	Output 0	OFF
Input 1	OFF	0	Output 1	OFF

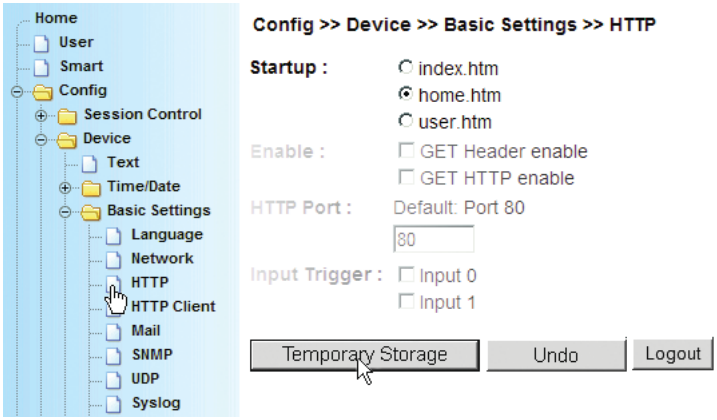
Alarme			
Alarm	*Status*	*Alarm*	*Status*
1	---	3	---
2	---	4	---

Time Switch Preview

To be able to operate the outputs as well it is necessary to log in to the Web-IO from the navigation tree as an administrator or with operator rights. The factory default setting of the Web-IO is for the Home page to be visible next to the navigation tree. To display only the Home page when opened in the browser there are two possibilities:

1. Entry in the address line http://<ip-address>/home
2. Configure the Web-IO so that only the Home page is displayed.

To do this, log in as administrator and under *Config >> Device >> Basic Settings >> HTTP* select *home.htm* at *Start Up*.



All additional settings on this page are described in the manual and should not be changed at this point.

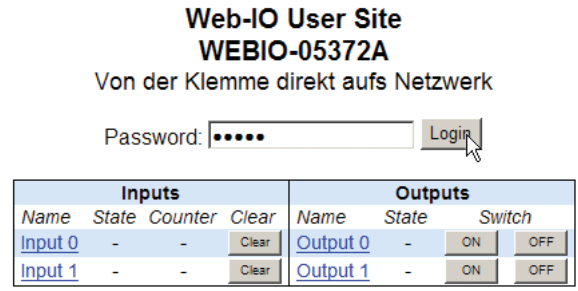
The Home page can be replaced with an individually constructed Web page. For details, see section „Access from your own application“.

5.1.2 The User page

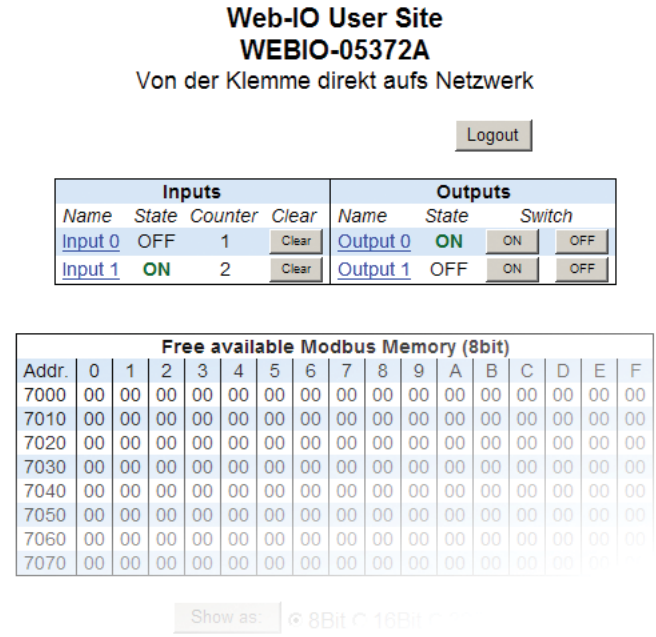
The User page shows the status of inputs, counters and outputs. In contrast to the Home page you can log in for the functions of this page without accessing the navigation tree. A password entered at login is stored on the PC as a cookie.

For the user the advantage of this is that he can access the outputs from the same PC when repeatedly opening the User page without having to log in each time, since login is automatic.

If a password was assigned for the Web-IO, login is required for displaying the status of inputs, outputs and counters!



After entering the password the status of the IOs is displayed and the outputs can be operated.

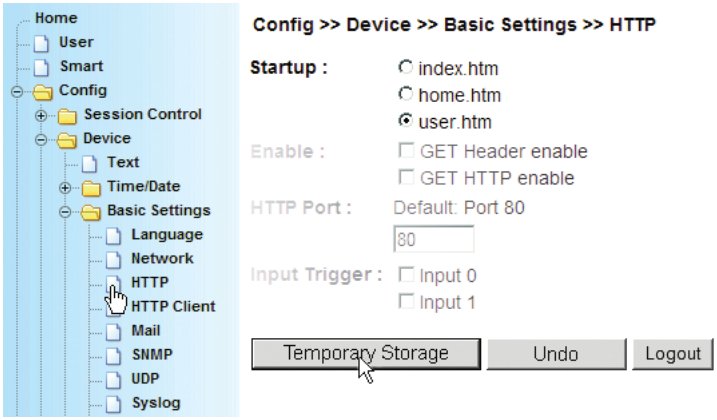


Only if the Web-IO has been configured for Modbus access does a table also appear beneath the IO status in which the contents of the available Modbus memory is displayed. (For details see section „Integration into existing systems >> Modbus“)

There are two ways to display only the User page without the navigation tree when opening in the browser:

1. Entry in the address line http://<ip-address>/user
2. Configure the Web-IO so that only the Home page is displayed.

   To do this, log in as administrator and under *Config >> Device >> Basic Settings >> HTTP* select *user.htm* at *Start Up*.



All additional settings on this page are described in the manual and should not be changed at this point.

The User page can be replaced with an individually constructed Web page. For details, see section „Access from your own application".

5.1.3 The Smart page

The Smart page has been optimized for display on smart phones such as the iPhone and is opened at http://<IP-address>/smart.

The Smart page displays the status of inputs, counters and outputs. Just as with the User page login is possible for the page functions without also accessing the navigation tree. A password entered at login is stored on the smart phone as a cookie.

For the user the advantage of this is that he can access the outputs from his smart phone when repeatedly opening the Smart Phone page without having to log in each time, since login is automatic..

If a password was assigned for the Web-IO, login is required for displaying the status of inputs, outputs and counters!

5.1.4 Labeling and texts

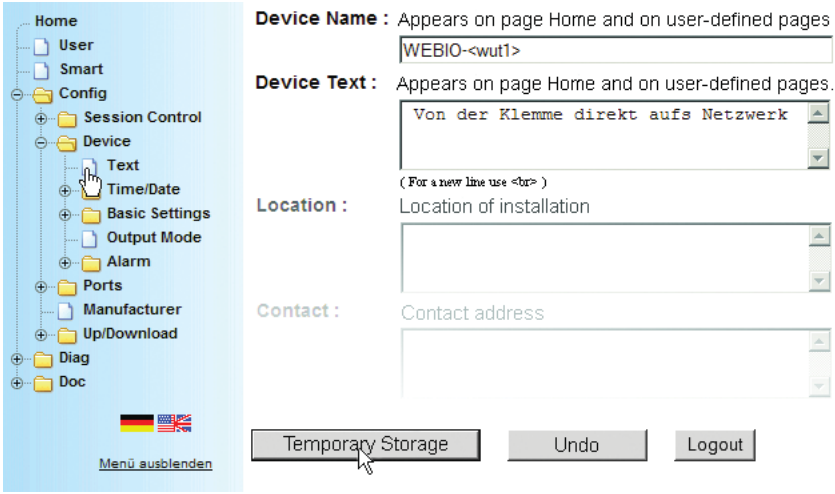
Required access rights: *Administrator*

The appearance of the Web-IO standard pages in the browser can be individually designed and modified in that naming of

the Web-IO, the inputs, counters and outputs is freely configurable.

Device name and description

You can change the device name and description of the Web-IO under *Config >> Device >> Text*.



If the description extends over more than two lines, the <br> tag is inserted at the position desired for the line break.

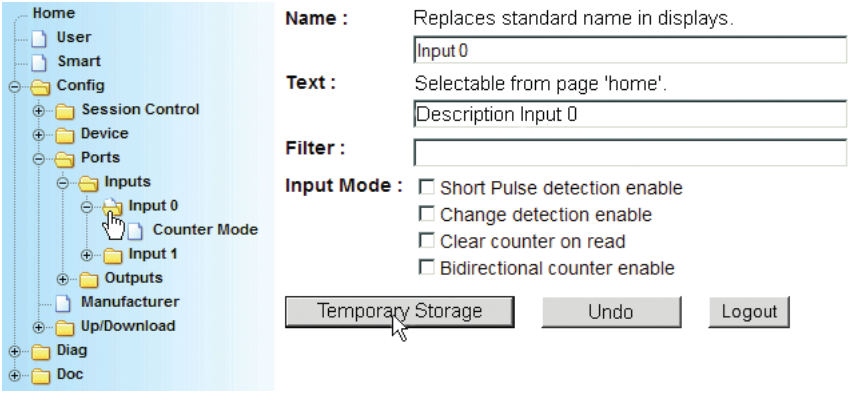
The *Location* and *Contact* fields are not displayed in the browser and are used only for identifying the Web-IO.

The display in the browser changes are *Apply* and *Logout > Save*.

Port name and description

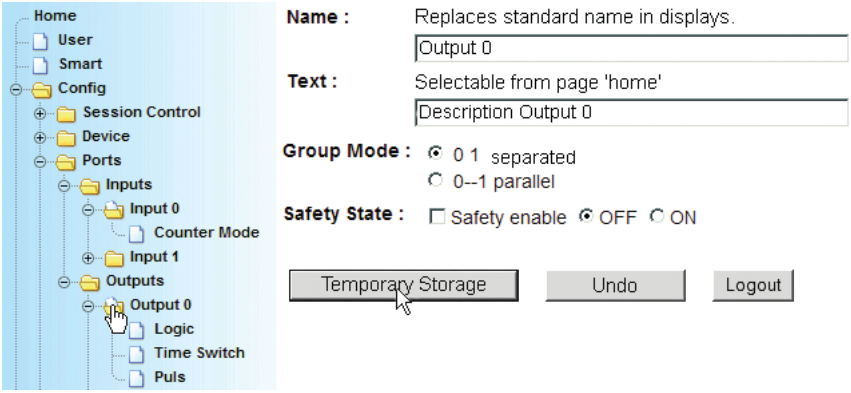
Individualized names and descriptions can also be created for the inputs and outputs.

To change the input name use *Config >> Ports >> Inputs* and for Input 0 *>> Input 0*:



The settings for Filters *and* Input Mode *should not be changed. For detailed information about these parameters see section* „Port Settings“

To change the output names select *Config >> Ports >> Outputs* and e.g. *>> Output 0* for Output 0

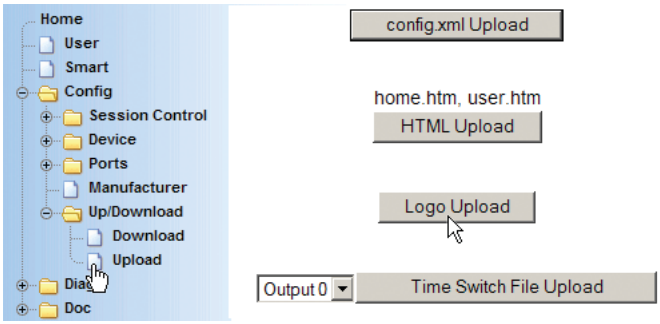


The settings for Group Mode *and* Safety State *should not be changed. For detailed information about these parameters see section* „Port Settings““

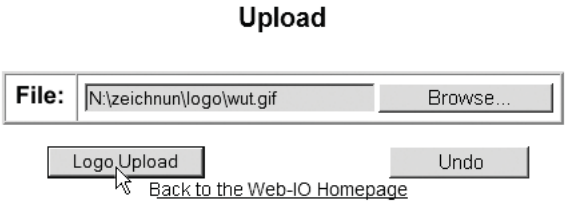
The display in the browser changes after *Apply* and *Logout > Save*.

Logo in the navigation frame

To enable as custom an appearance of the Web-IO Start page as possible, it is possible to replace the W&T logo in the navigation frame with your own. In the navigation tree and select *Config >> Up/Download >> Upload*



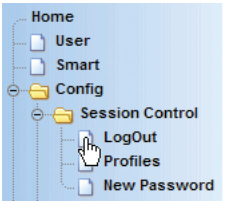
Click on the *logo.gif upload* button.

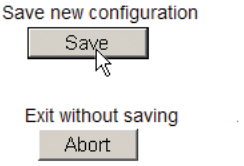


Select the desired JPG or GIF file and click on the *logo.gif upload* button.

Note that the dimensions of the logo should be close to those of the W&T logo (width max. 200 pixels, height max. 100 pixels) so that the clarity of the navigation frame is retained.

The display in the browser changes after *Apply* and *Logout > Save*.





If you do not want the new logo to appear right away the next time the system is started up, delete the browser cache!

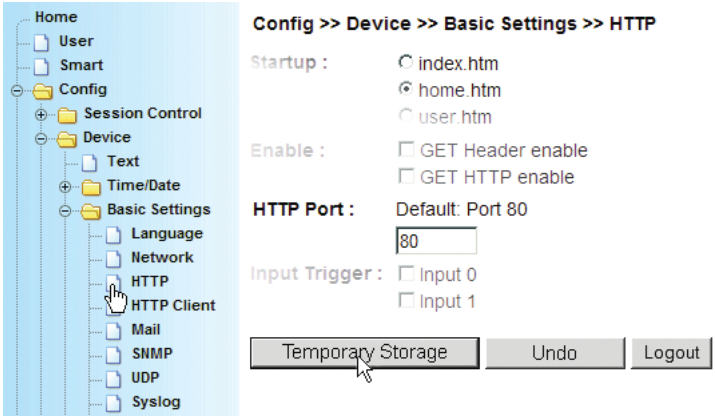
5.1.5 Changing the HTTP port

Browser access to the Web server of the Web-IO is made according to HTTP protocol using TCP Port 80 (factory default setting).

There are cases such as NAT routing or port forwarding which make it necessary to configure a different port for HTTP access.

To do this, log in as administrator and in the navigation tree and select *Config >> Device >> Basic Settings >> HTTP.*

Under *HTTP Port* you can enter a TCP port for HTTP access which differs from the standard Port 80.



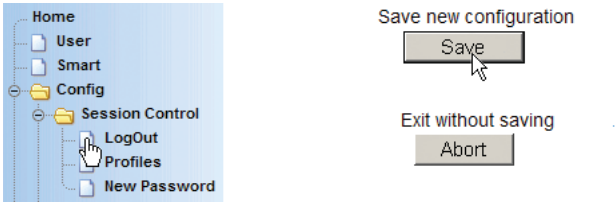
All other settings on this page are described later in the manual and should not be changed at this point.

If the HTTP port is changed, the Web-IO can be reached from the browser only by additionally specifying the selected HTTP port.

Example for HTTP Port = 85.

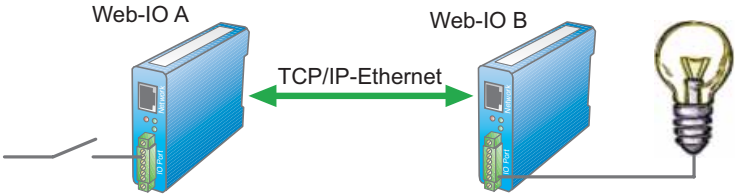
This also applies to all configuration access.

The change takes effect after *Apply* and *Logout* > *Save*.



In this mode the inputs of a Web-IO Digital are sent to the outputs of a second Web-IO and vice-versa.

5.2 Box-to-Box



In the case of Box-to-Box connections one Web-IO assumes the function of the master.

The second Web-IO functions as a slave. The slave waits for the master to connect with it.

Both the master and the slave must be configured correspondingly.

For Web-IOs with relay outputs you must maintain a pause of at least 200ms between two switching operations on one output!

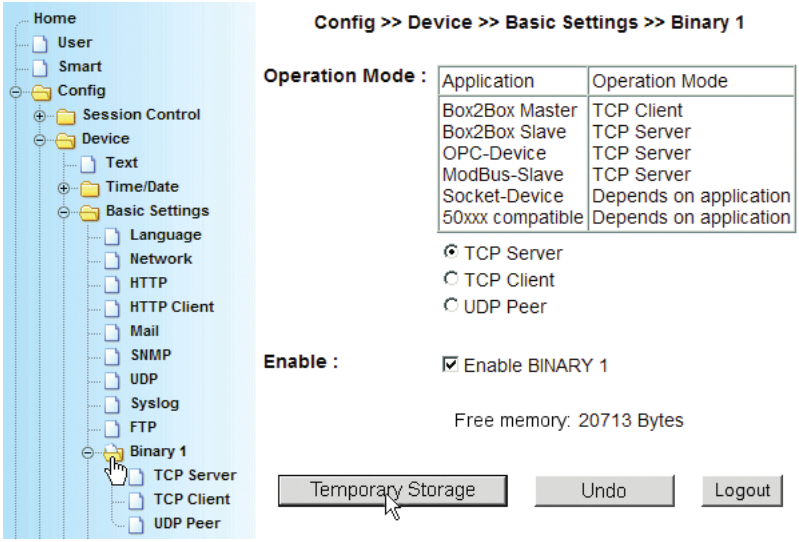
5.2.1 Configuring the Slave Web-IO

Required access rights: *Administrator*

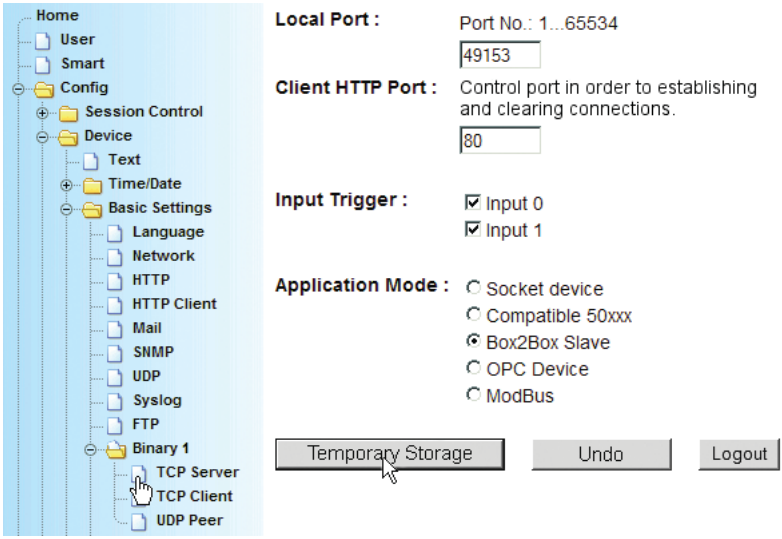
Select in the navigation tree des Slave Web-IO *Config >> Device >> Basic Settings >> Binary 1*

Set *TCP-Server* as the *Operation Mode*.

Then click on *Temporary Storage* to send the changes to the Web-IO.



Now select: *Config >> Device >> Basic Settings >> Binary1 >> TCP-Server.*



Local Port:

Unless your network administrator has given you other instructions, the factory default port 49153 can be used.

A reason for changing the factory default Local Port might be for example a firewall that only allows certain port accesses.

In any case the Local Port for the slave must be identical with the Server Port *entry of the master.*

Client HTTP Port
Specifies the HTTP port on which the control connection to the Master box should be opened.

Unless otherwise specified, always use Port 80.

Input Trigger:
Here you activate the inputs that you want to set the corresponding outputs on the master.

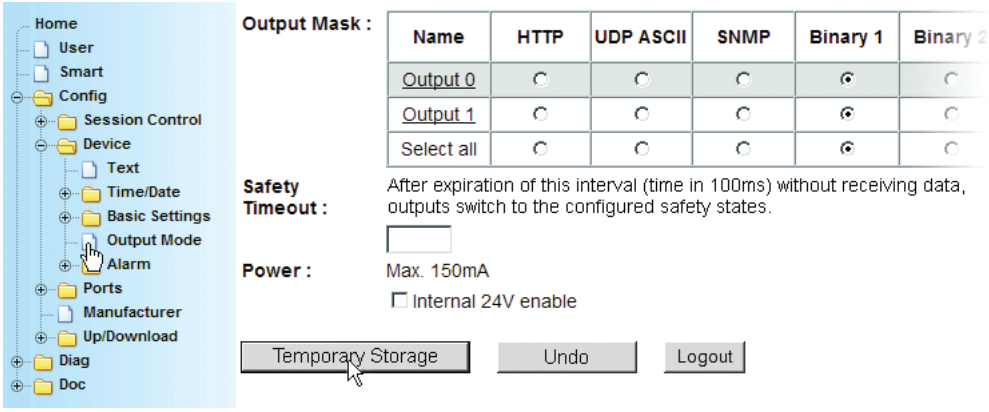
The Web-IO 12XDigital allows simultaneous access to the inputs from various operation modes.

This means for example that inputs 0-5 control outputs 0-5 on the master Web-IO, but can also be read via HTTP at the same time.

Application Mode
Select *Box2Box Slave*

After you have entered all the parameters, confirm by clicking on the *Temporary Storage* button.

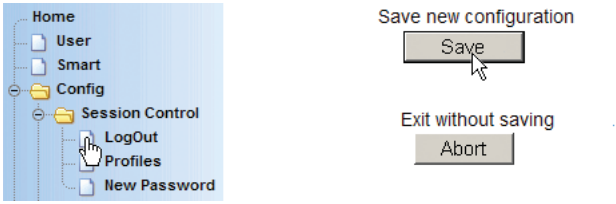
Select: *Config >> Device >> Output Mode*



Here you activate the outputs that will be set by the corresponding slave inputs for Binary 1 and confirm by clicking on *Temporary Storage.*

In contrast to the inputs, the outputs activated for Box-to-Box are no longer accessible for other operation modes.

Now you still need to activate the new settings. Use the *Logout* button or select *Config >> select >> Session Control >> LogOut.*



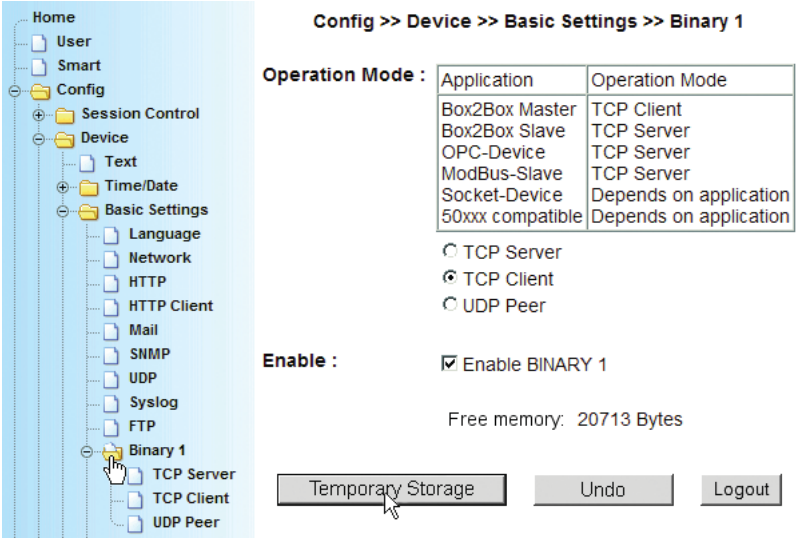
After clicking on *Save* all the settings are updated in the Web-IO and the start page is reopened in Default User mode. The master Web-IO can now connect to the slave Web-IO.

The Box-to-Box Master can only connect with the Slave if there is no Administrator or Operator login for *the Slave*

5.2.2 Configuring the master

Required access rights: *Administrator*

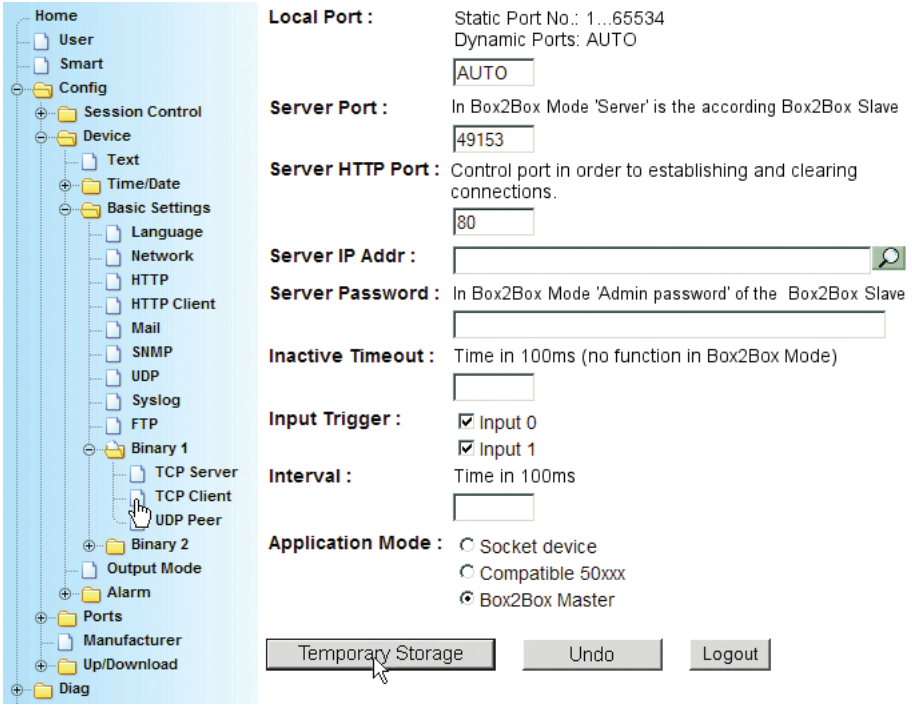
Select in the navigation tree: *Config >> Device >> Basic Settings >> Binary1*



Set *TCP-Client* as the *Operation Mode*.

Then click on the *Temporary Storage* button to send the changes to the Web-IO.

Select in the navigation tree: *Config >> Device >> Basic Settings >> Binary1 >> TCP-Client.*



The following parameters must be entered:

Local Port:
Unless your network administrator has given you other instructions, the factory default Port 49153 can be used.

ServerPort:
Here you must enter the *Local Port* set for the slave. Here again you can use the factory default 49153 if your network administrator has not specified otherwise.

Local Port and Slave Port do not necessarily have to be the same as set at the factory.

A reason for changing the factory default values for *Local Port* and *Slave Port* might be for example a firewall that only allows certain port accesses.

Server HTTP Port
Specifies the HTTP port on which the control connection to the Salve should be opened.

Unless otherwise specified, always use Port 80.

Server IP Addr:
Enter here the IP address of the Web-IO that you want to serve as a slave.

Server Password:
Here you enter the Administrator password for the slave Web-IO. If no password was assigned for the slave, leave this field blank.

Inactive Timeout
This parameter has no function in Box-to-Box mode, since a continuous connection is desired.

Input Trigger:
Activate here the inputs that will set the corresponding slave outputs.

The Web-IO 12XDigital allows simultaneous access to the inputs from various operation modes.

This means for example that inputs 0-5 control outputs 0-5 on the slave Web-IO, but can also be read via HTTP at the same time.

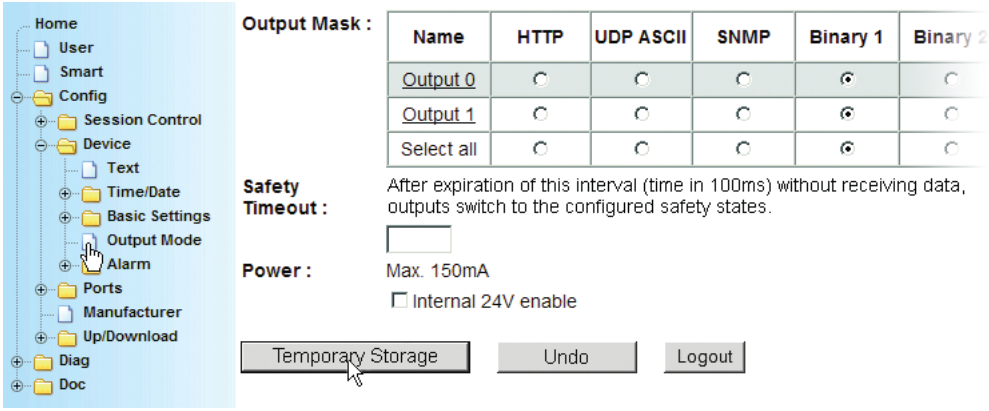
Interval:
If no interval is entered, the state of the inputs is sent to the outputs of the other Box-to-Box partner each time the state changes. Entering an interval sends the state cyclically even if there is no change.

If two locations are connected over a fee-based ISDN line, you should not use an interval, since depending on the timeout and interval the ISDN connection is never closed or is very frequently reopened.

Application Mode

Select *Box2Box Master*
After you have entered all the parameters, click on the *Temporary Storage* button.

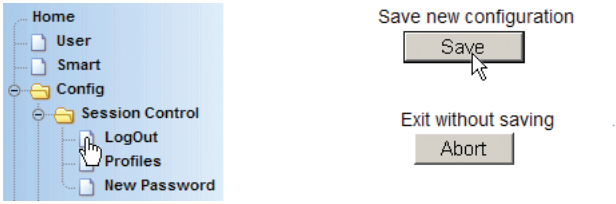
Select in the navigation tree: *Config >> Device >> Output Mode*



Here you activate the outputs that will be set by the corresponding slave inputs for Binary 1 and confirm by clicking on the *Temporary Storage* button.

In contrast to the inputs, the outputs activated for Box-to-Box are no longer accessible by other operation modes.

Now you must activate the new settings. Use the *Logout* button or *Config >> select >> Session Control >> LogOut.*



After clicking on the *Save* button all the settings are updated in the Web-IO and the start page is reopened in Default User

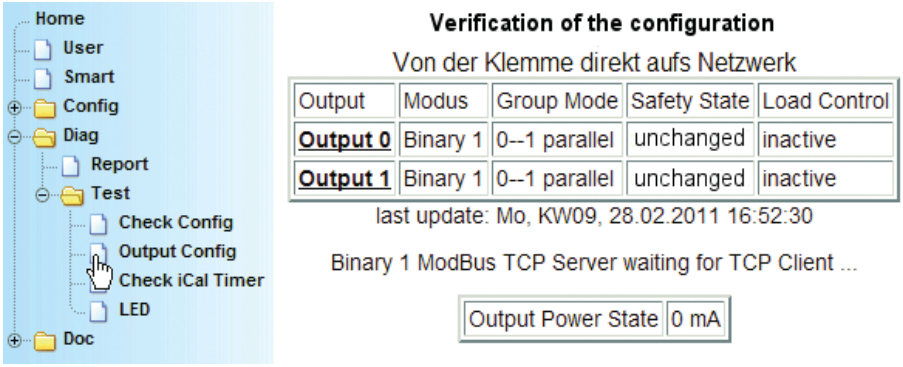
mode. The master Web-IO then attempts to open a connection to the slave Web-IO.

All the functions described here for Binary 1 can of course also be used under Binary 2. Thus a Web-IO A can be configured in the Binary 1 area so that the inputs/outputs 0-5 Box-to-Box work with a Web-IO B. In the Binary 2 area inputs/outputs 6 - 11 can then be configured so that they work Box-to-Box with another Web-IO.

5.2.3 GetBox-to-Box connection status

Required access rights: *Administrator*

The connection status of a Box-to-Box connection can be obtained from the navigation tree under *Diag >> Test >> Output Config*.



Here you are shown which operation mode the individual outputs are working in. Additionally the current status of a Box-to-Box connection is displayed at the bottom of the Web page.

5.2.4 Stopping Box-to-Box mode

Box-to-Box Modus only by Master

Required access rights: *Administrator*

Stopping Box-to-Box mode should always be done using the corresponding configuration of the master. The master and slave Web-IO must be connected on the network. Select in the navigation tree of the master : *Config >> Device >> Basic Settings >> Binary1 >> TCP Client* and delete the entry for *Server IP Addr*. Also set *Application Mode* to *Default*.

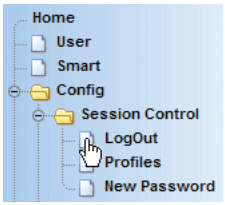
Confirm by clicking on the *Temporary Storage* button.

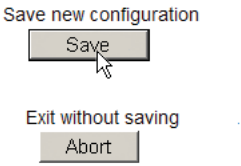
Then under *Config >> Device >> Basic Settings >> Binary1 >>* set the *Operation Mode* to *TCP Server* .

Confirm by clicking on the *Temporary Storage* button.

Now select in the navigation tree of the master : *Config >> Device >> Output Mode* and set the outputs that you no longer want to work under Box-to-Box to HTTP.

Confirm by clicking on the *Temporary Storage* button. Now you must still activate the new settings. Use the *Logout* button or in the Config menu *select >> Session Control >> LogOut.*





After clicking on the *Save button* all the settings are updated in the Web-IO and the start page is reopened in Default User mode.

Stopping Box-to-Box Modus for the Slave Web-IO

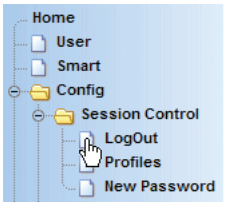
Required access rights: *Administrator*

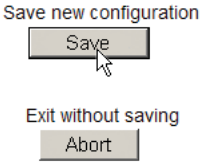
Select in the navigation tree of the slave: *Config >> Device >> Basic Settings >> Binary1 >> TCP Server* and set *Application Mode* to *Default.*

Confirm by clicking on the *Temporary Storage* button.

Now select in the navigation tree *Config >> Device >> Output Mode* and set the outputs that you no longer want to work under Box-to-Box to HTTP.

Confirm by clicking on the *Temporary Storage* button. Now you must still activate the new settings. Use the *Logout* button or In the Config menu *select >> Session Control >> LogOut.*



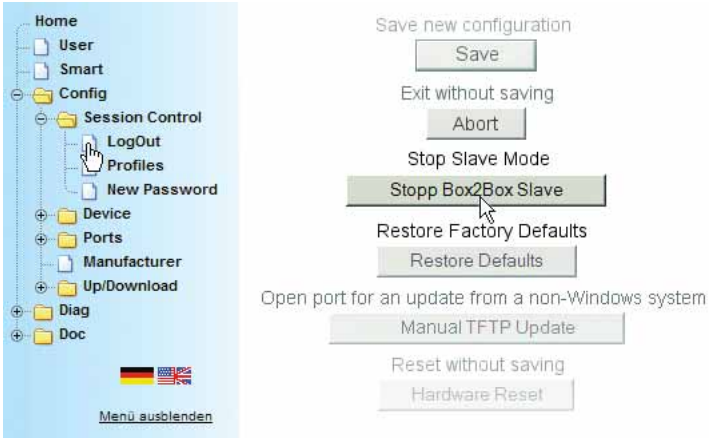


After clicking on the *Save* button all the settings are updated in the Web-IO and the start page is reopened in Default User mode.

5.2.5 Stopping Box-to-Box Modus only for the Slave

If the master is no longer available, for example because there is no network connection but you still want to deactivate Box-to-Box mode for the slave, select in the navigation tree *Config >> Session Control >> LogOut.*

In the configuration frame there should be an additional button *Stop Box2Box Slave.*



If this button is not displayed, first click on the *Reset* button. This will restart the Web-IO.

After a new Login and invoking *Config >> Session Control >> LogOut* the *Stopp Box2Box Slave* button will be displayed. Clicking on it will deactivate Box-to-Box mode in the slave.

5.2.6 Box-to-Box notes and limitations

Runtimes
Please note that when sending data using a network protocol, system-related runtime delays occur.

In contrast to a direct cable connection, whereby a switching signal is present directly on the actuator, latency times of between 20 and 50 ms must be factored in.

If active components such as routers, bridges or even ISDN segments are included in the transmission path, the latency time increases correspondingly, so that latency times of 500ms or more can occur.

Box-toBox with multiple partners
It is not possible to outputs signals from an input to the outputs of multiple Web-IOs. It is possible however to use Bina-

ry 1 and Binary 2 modes of a master Web-IO for example to link inputs 0 - 5 to Slave A by tinputs 6 - 11 to Slave B.

Mixing the various Web-IO models
It is possible to connect different Web-IO models with each other box-to-box. The condition is that the Web-IO is configured with a smaller number of ports than the Master.

1. Example:
You want to connect a Web-IO 2xDigital to a Web-IO 12xDigital Box-to-Box.

The Web-IO 12xDigital in this case is configured under Binary 1 as Box-to-Box Slave, whereby the input triggers for Input0 and Input1 are set. Also, in the Output Mode area the outputs 0 and 1 are enabled for Binary 1.

The configuration is then saved.

The Web-IO 2xDigital is configured as a normal Box-to-Box Master for Binary 1.

2. Example
You want to connect a 2 Web-IO 2xDigitals to a Web-IO 12xDigital Box-to-Box.

As in the first example, you first configure the Web-IO 12xDigital as a Slave. This is done both for Binary 1 and Binary 2. The input triggers 0 and 1 are set for Binary 1., and input triggers 2 and 3 for Binary 2. In addition, Outputs 0 and 1 are configured for Binary 1 and Outputs 2 and 3 for Binary 2 in the Output Mode area.

The configuration is then saved.

The 1st Web-IO 2xDigital is configured as a normal Box-to-Box Master for Binary 1. Proceed similarly for the 2nd Web-IO 2xDigital for Binary 2.

After saving the settings for all Web-IOs involved, Output 0 on the Web-IO 12xDigital follows changes on Input 0 of the 1st

Web-IO 2xDigital. If Input 1 on the 2nd Web-IO 2xDigital changes, this results in a change on Output 2 of the Web-IO 12xDigital.

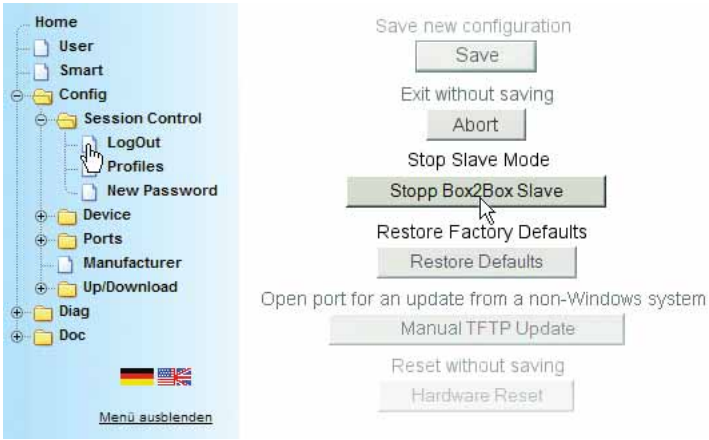
Old and new units in Box-to-Box combination

It is not possible to operate older 50xxx I/O Com-Servers with a Web-IO in Box-to-Box. Web-IOs working in Box-to-Box mode should have the same firmware version.

Problems with Box-to-Box

Especially when the IP addresses of a Box-to-Box pair have changed for example by a change in the infrastructure, it can happen that the Master can no longer open a connection to the Salve or vice-versa.

In such cases there should be an additional *Stopp Box2Box Slave* button under *Config >> Session Control >> LogOut*.



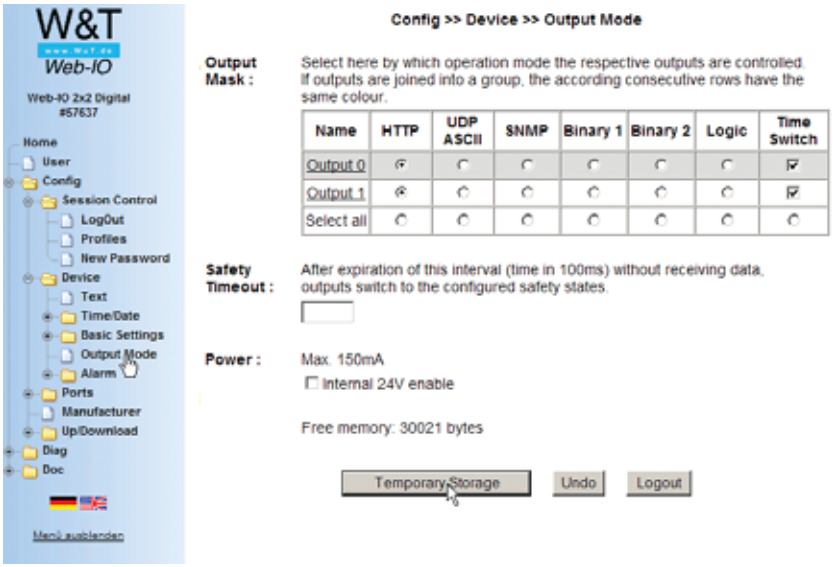
Clicking on this button resets any existing old connection parameters and opens the Slave box for a new connection.

5.3 Time Switch - Using the Web-IO as a Time Switch

The Web-IO Model 57637, in addition to its normal range of functions, can also be used to switch the outputs on a timed basis.

5.3.1 Activate the outputs for timed switching

From the Web-IO navigation tree select *Config >> Device >> Output Mode*



Here you activate the outputs you want to switch on a timed basis and confirm by clicking on the *Apply* button.

When the Web-IO is operating in *Output Mode Time* Switch, it can still be switched using the preset normal operating mode.

This means that the output state set by the time switch can still be changed, for example from the browser. The next timed switching action does then change the manually set state.

Parallel accessing of Logic and Time Switch is not permitted.

5.3.2 Load the switching defaults

The Web-IO does not have a configuration interface for setting the time switch. Instead there are two ways of loading the control data into the Web-IO in the form of an iCal calendar file:

- Load the iCal file from an online calendar directly over the Internet
- Load the iCal file from the local PC

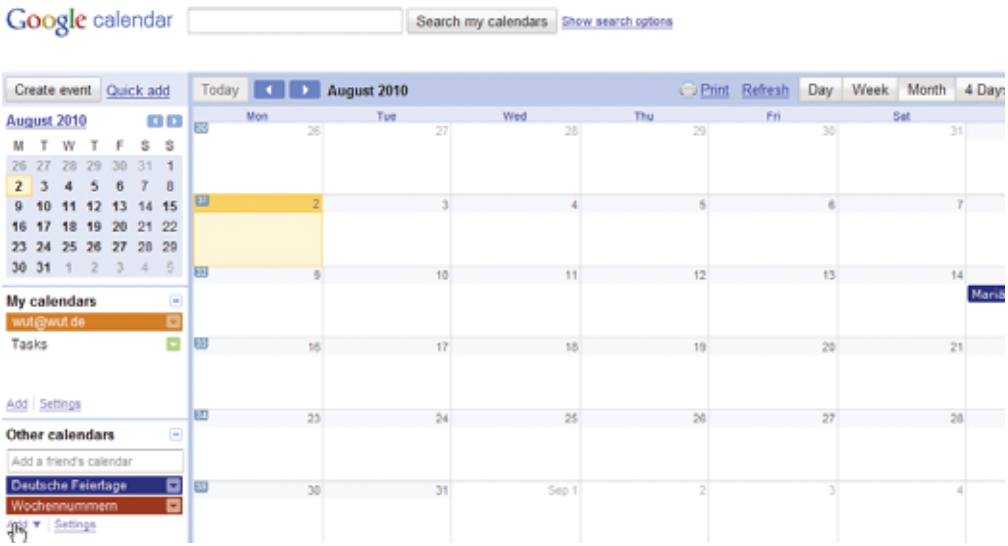
5.3.3 Load switching times from an online calendar

Here we show an example of how to manage the switching times using the Google calendar. To use the functionality of the Google calendar, you must set up a corresponding user account at www.google.com.

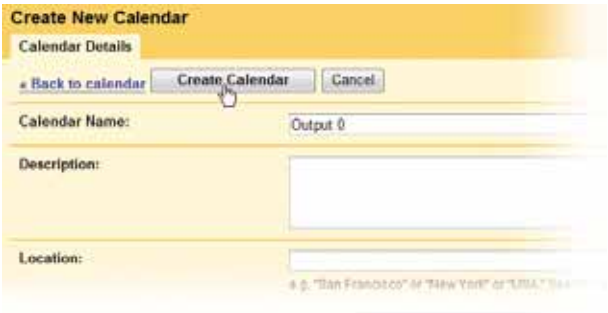
Please note the terms of use for the Google account which you agree to when setting up the account..

You may of course use any other online calendar which provides a public link to the iCal file associated with the calendar.

Each of the two outputs requires its own calendar in which you enter the switching defaults. Google allows multiple calendars to be maintained under a single account.



To create a new calendar, select *Add* under *My calendars*

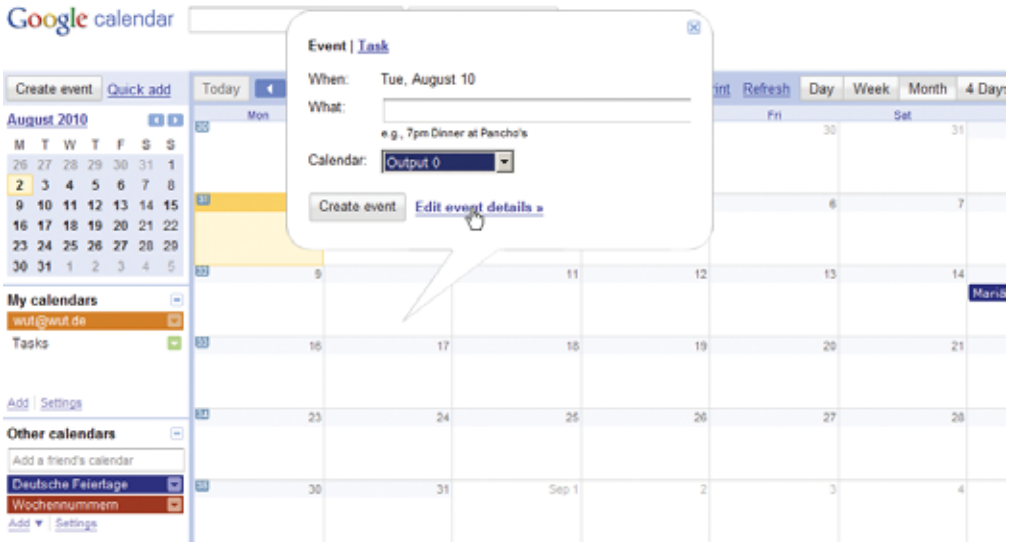


It makes no difference what names you assign to the calendars. We do however recommend giving a meaningful name so that it is easier to distinguish them when using multiple Web-IOs.

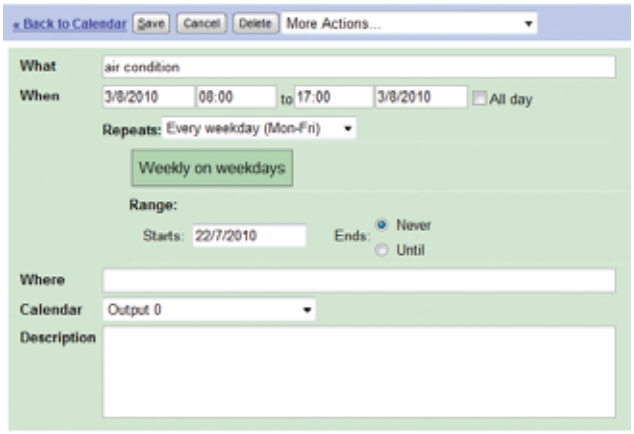
No other entries or changes are necessary at this point. To create the calendar, simply click on *Create calendar.*

If one calendar has been created for both outputs, you can enter the desired switching times.

The Google calendar permits both one-time and recurring events. To add a new event, click on *Create event* or on the desired day in the calendar.



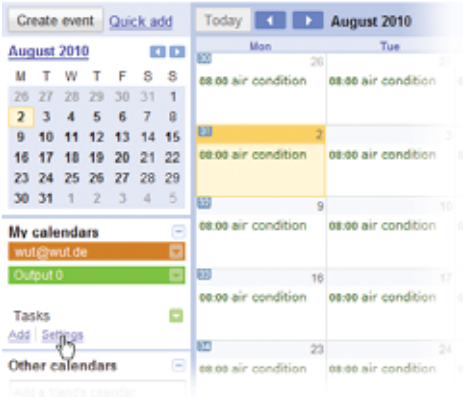
For *Calendar* select the calendar for the corresponding event and then click on *Edit Edit event details.*



Enter all the event details and then click on *Save*. It makes no difference which name you enter under „What:“ The Web-IO

switches the output on at the beginning of each found event and off at the end of the event.

Use this procedure to configure all the desired switching times.

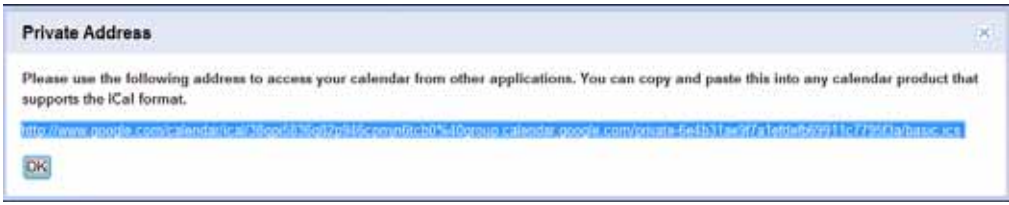


Finally, under *My calendars* click on the *Settings* link.

Select the calendar for Output 0.

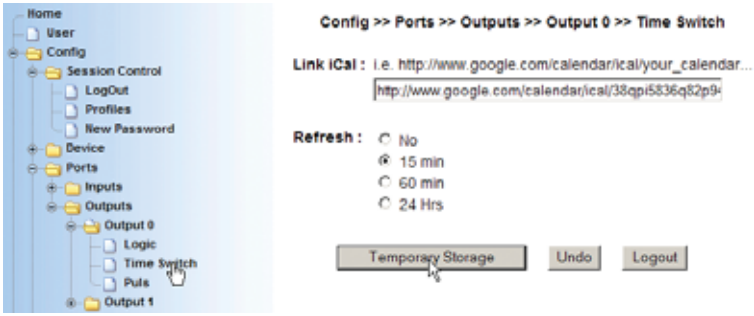
Calendar Address:
Learn more
Change sharing settings
XML ICAL HTML (Calendar ID: 38qpi5836q82p946cpmjn6tcb0@
This is the address for your calendar. No one can use this link unless you
Private Address:
Learn more
XML ICAL HTML Reset Private URLs
This is the private address for this calendar. Don't share this address with
calendar.

In the lower section of the resulting Web page you will find the item *Private address*. Click here on *iCal*.


Private Address
Please use the following address to access your calendar from other applications. You can copy and paste this into any calendar product that supports the iCal format.
OK

Next the online link to the iCal file associated with the calendar is shown. Select this link and copy it to the clipboard using *<Ctrl> - C* .

Now in the navigation tree of the Web-IO select *Config >> Ports >> Outputs >> Output 0 >> Time Switch.*



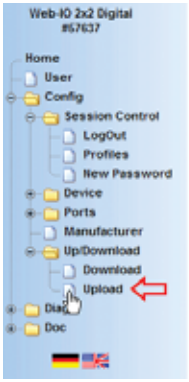
Link iCal

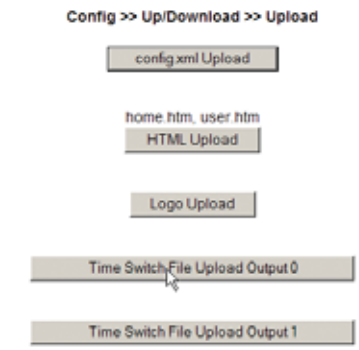
Use *<Ctrl> - V* to paste the link that was copied to the clipboard.

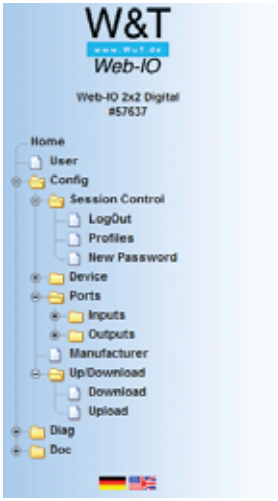
Refresh

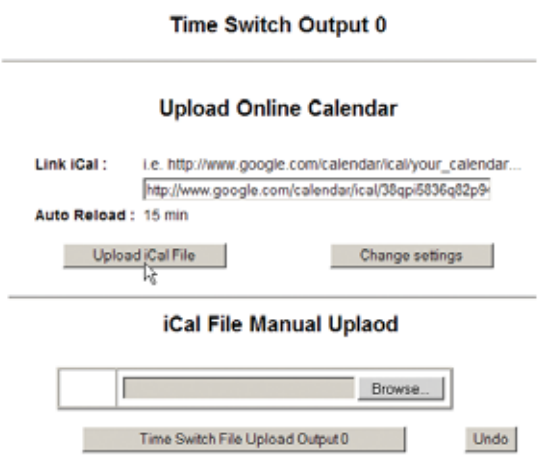
Here you determine the cycle at which the Web-IO downloads the calendar data from the online calendar. Checking *No* means no automatic refreshing takes place, and the file must be downloaded manually.

To do this, go to the navigation tree for the Web-IO and select *Config >> Up/Download >> Upload* and then click on *Time Switch File Upload Output 0.*





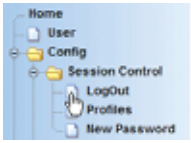




Click on *Upload iCal Datei Output 0*.

Use the same procedure for Output 1.

Now you must activate the new settings. Use the *Logout* button or *Config >> Session Control >> LogOut*.



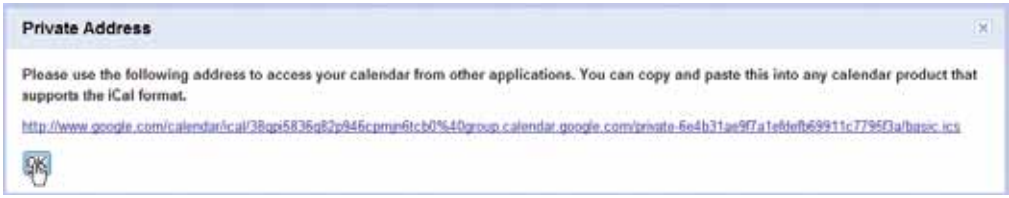
5.3.4 Load switching times as a iCal file

If the Web-IO does not have access to the Internet, you can instead load the switching times into theunit from the hard drive of a local PC in the form of an iCal file.

Here again you can create the switching times using an online calendar. The procedure is identical to that described previously.

Note however that the iCal file must first be downloaded to a PC connected to the Internet.

In the case of the Google calendar you only need to click on the link for the iCal file.


Private Address
Please use the following address to access your calendar from other applications. You can copy and paste this into any calendar product that supports the iCal format.
http://www.google.com/calendar/ical/38qpi5836q82p946cpmn6tcb0%40group.calendar.google.com/private-6e4b31ae9f7a1e8defb69911c7795f3a/basic.ics
OK

The downloaded iCal file can then be transferred to a PC connected to the Web-IO using a memory stick.

Of course the iCal file can also be created using any calendar program having an iCal export function.

A recommended freeware program is Mozilla's Sunbird.

Free download at:

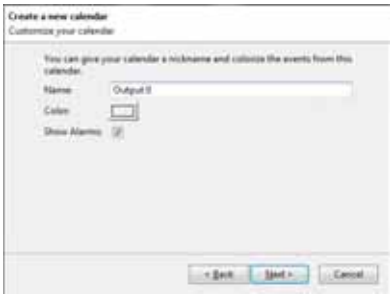
http://www.mozilla.org/projects/calendar/sunbird/

The installation for Sunbird is self-explanatory.

After starting up for the first time, select *File >> New calendar.*

Create a new calendar
Locate your calendar
Your calendar can be stored on your computer or be stored on a server in order to access it remotely or share it with your friends or co-workers.
On My Computer
On the Network
< Back
Next >
Cancel

Save the calendar on the local PC.

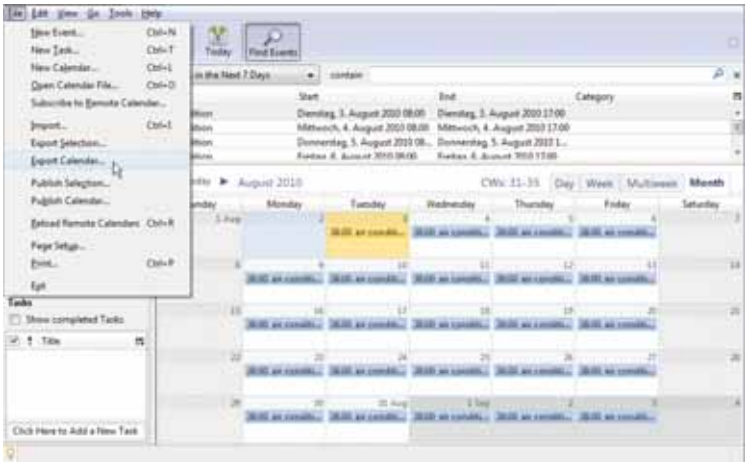


Give the calendar a name and confirm by clicking on *Next*.

Sunbird permits both single and recurring events. To create a new event, double-click on the desired day in the calendar.

Specify all the details of the event and then click on *Save* and *Close*. It makes no difference what name you enter under *Title*. The Web-IO will switch the output on at the beginning of each found event and off at the end of the event.

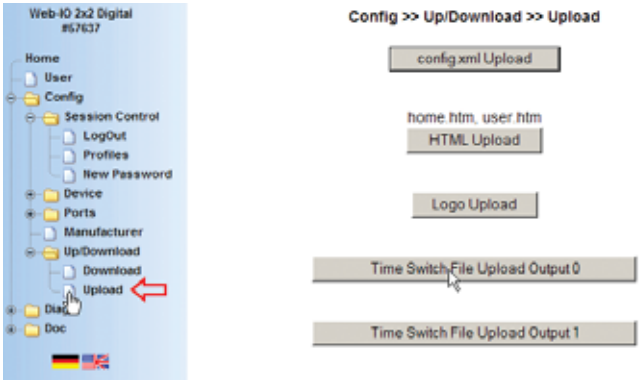
Use this procedure for configuring all your desired switching times.



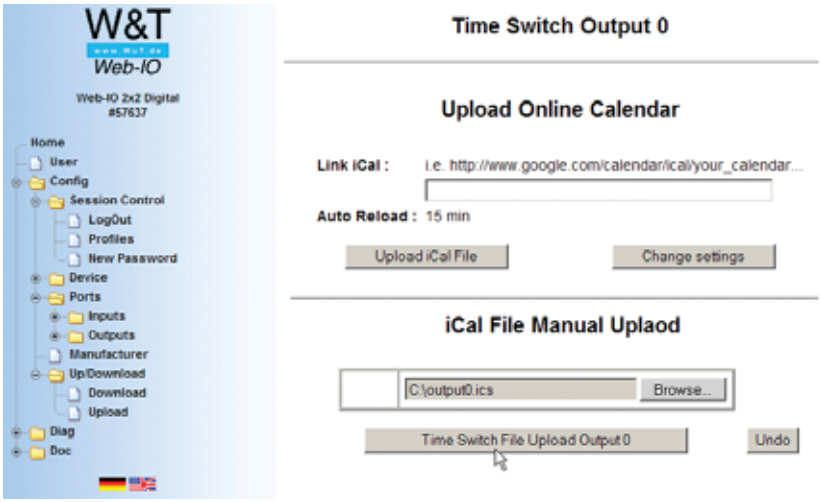
Then under *Calendar* click on *Export file*.

Save the iCal file on the hard drive of the PC.

In the Web-IO navigation tree select *Config >> Up/Download >> Upload* and then click on *Time Switch File Upload Output 0*.



To load the iCal file into the Web-IO, select the desired file and click on *Time Switch File Upload Output 0*.



Proceed in the same manner for Output 1.

Now you must still activate the new settings. Use the *Logout* button or select *Config >> Session Control >> LogOut.*

5.3.5 Time Switch preview

After you have finished configuring the Time Switch the Web-IO begins to run the loaded time control program.

You can open a preview of the switching events of the next 24 hours from the homepage of the Web-IO.

To do this, simply click on *Time Switch Preview.*

Preview iCal Timer
WEBIO2x2Digital-04D9D7

Time Switch Preview

Date	Time	Output 0	Output 1
02.06.2010	12:18:24	ON	OFF
02.06.2010	17:00:00	OFF	OFF
02.06.2010	21:00:00	OFF	ON
03.06.2010	06:00:00	OFF	OFF
03.06.2010	08:00:00	ON	OFF

5.3.6 Special features of the Web-IO Time Switch

It is possible to both configure the refresh cycle with an online calendar and upload an iCal file.

In practice this makes sense only of you need to switch temporarily at times other than what you have set for normal operation.

If a cycle has been configured for the onloine refresh, the manually uploaded switching times apply only until the next cycled online refresh takes place.Then the switching times from the online calendar are applied again.

Regardless of which way the Web-IO has received its time switching defaults, if in addition to Time Switch one of the basic operating mode types

- HTTP
- UDP ASCII
- SNMP
- Binary 1
- Binary 2

is activated from *Config, Device >> Outputmode*, the current switching status can be changed at any time using the set access type. The next timed switching action would however again determine the output status.

5.4 Integration into existing systems

To make it easy to integrate the Web-IO into existing systems, it also supports several common standard communication methods used in automation and network technology:

- SNMP
- OPC
- Modbus

5.4.1 SNMP - In-/Outputs in management systems

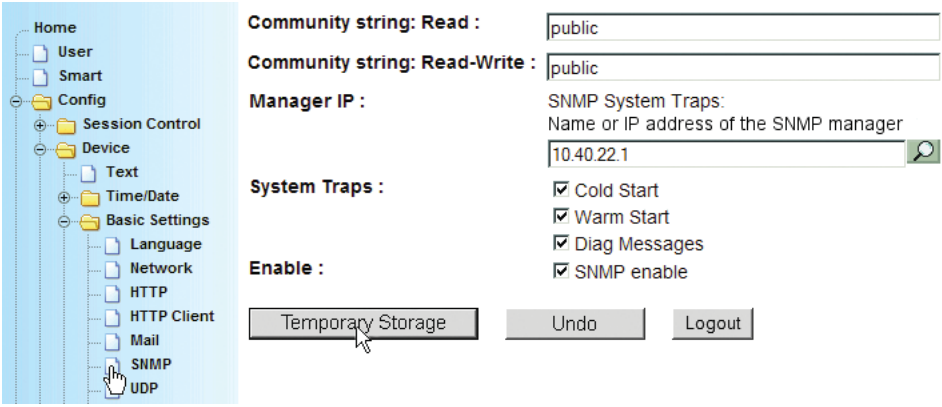
The Web-IO Digital can be incorporated into existing network management systems.

Here the entire configuration as well as the controlling and monitoring of inputs, counters and outputs can be handled with SNMP.

The Private-MIB is available at http://www.wut.de for for downloading.
From there the easiest way to navigate is using the menu tree on the life side. The path Products & Downloads >> Web-IO >> Tools takes you to the Web page containing a direct link to the MIB download.

Enabling SNMP access
Required access rights: *Administrator*

Select in the navigation tree des Web-IO *Config >> Device >> Basic Settings >> SNMP*



Community String: Read
Enter here the community string for SNMP access with read authorization.

Community String: Read-Write
Enter here the community string for SNMP access with read and write authorization.

Manager IP:
IP address of the SNMP Management-System.

System Traps:
The Web-IO offers the possibility of sending a system trap to the entered SNMP manager at a cold or warm start (Hardware-Reset and Power-ON or Software-Reset).

Select *Cold Start* or *Warm Start*.

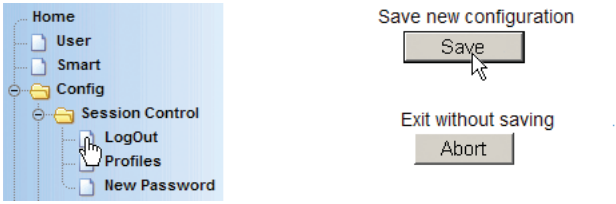
Diag Messages:
This option can be used to send error and diagnostic messages as an SNMP trap to the SNMP management system. As long as the internal memory of the Web-IO is not deleted, an error is sent as a trap only when it first occurs. Which errors are currently present can be seen by opening the Diag page in the browser.

See: http://<IP address of the Web-IO/diag

Enable:
At this point SNMP Enable must be activated in order to work with the Web-IO on SNMP level.

Confirm your entries by clicking on the *Temporary Storage* button.

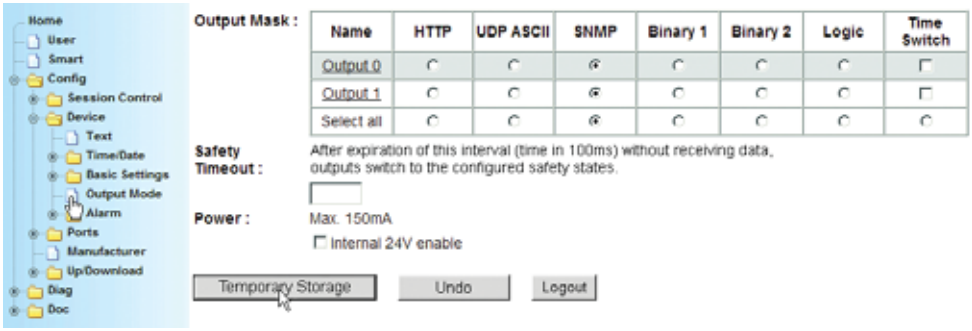
If you have no more configuration changes to make, you still must activate the new settings. Use the *Logout* button or select *Config >> Session Control >> LogOut.*



After clicking on the *Save* button all settings in the Web-IO are updated and the start page is restructured in Default User mode.

Activating the outputs for SNMP
Required access rights: *Administrator*

Select in the navigation tree of the Web-IO *Config >> Device >> Output Mode*



Activate here the outputs you want to control using SNMP protocol and confirm by clicking on the*Temporary Storage* button.

Now you must activate the new settings. Use the *Logout* button or In the Config menu *select >> Session Control >> LogOut.*

After clicking on the *Save* button all the settings are updated in the Web-IO and the start page is reopened in Default User mode.

Main steps for SNMP access
The private MIB provided by W&T can be used for all Web-IO Digital devices. Separate OID branches are used for the various models

The procedure shown here applies as an example for the Web-IO 12xDigital Input, 12xDigital Output.

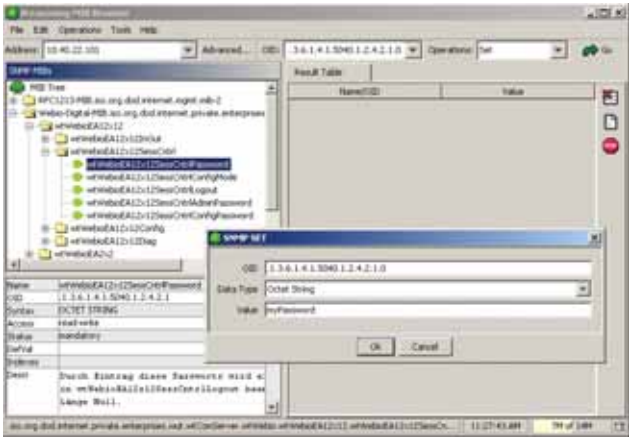
As soon as the SNMP access was enabled as described above, read access is possible at any time.

To have write access to the Web-IO, you must first open an SNMP session. This applies especially if you want to use SNMP for setting the outputs.

Session Login
To open a session, you must enter the Administrator password into the variable

`wtWebioEA12x12SessCntrlPassword`

An opened session remains viable for 5 minutes and is then automatically closed.



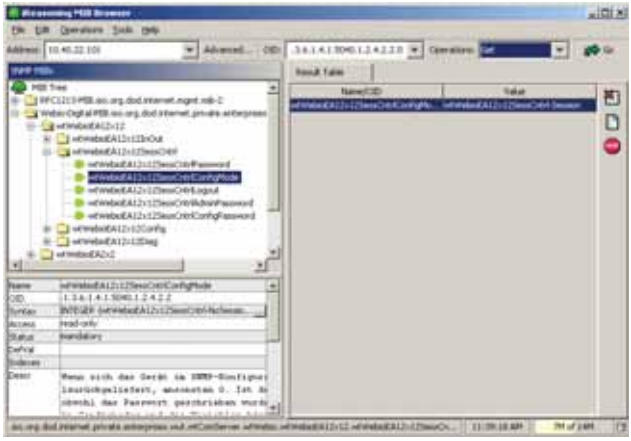
Checking a session for validity

By reading out the variable

`wtWebioEA12x12SessCntrlConfigMode`

you can check whether the session was successfully opened.

1 = Session open, Web-IO in Configuration Mode
0 = Opening of the session failed. Check whether the password was incorrectly entered.



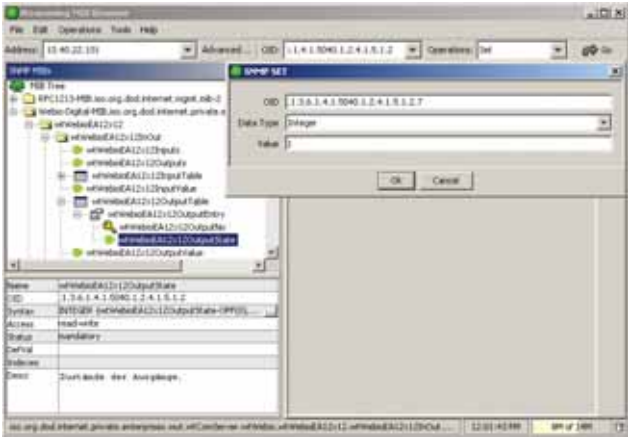
After successful opening of the session, the variables defined in the private MIB can be used to make any desired configuration changes or input and output accesses.

Changing an output

To change the state of individual outputs after a successful session login, use the SNMP variable

`wtWebioEA12x12OutputState`

To specify which output you want to change, the number of the output must be appended to the corresponding OID, separated by a decimal point.



A value of 1 for ON and 0 for OFF is used. Here for example Output 7 is set to ON.

Changing all outputs at the same time

In addition to setting individual outputs, you can change all outputs at once. To do this, use the SNMP variable

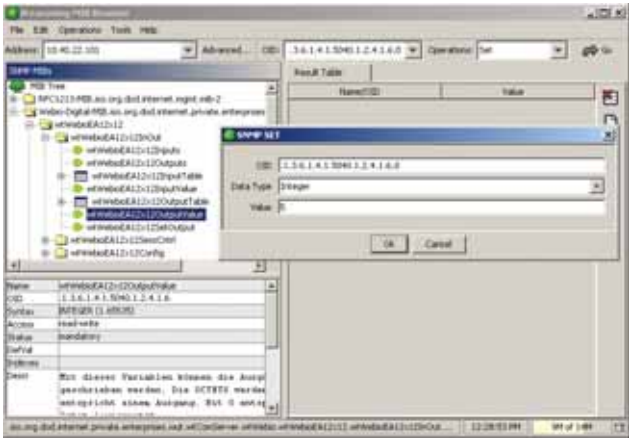
`wtWebioEA12x12OutputValue`

Sent is the integer value which corresponds to the binary pattern of the outputs.

Example: Set Output 0 and Output 2 to ON. All other outputs should assume the OFF state.

The value you send is calculated as follows:

$1x2^0 + 0x2^1 + 1x2^2 + 0x2^3 + 0x2^4 + 0x2^5 + 0x2^6 + 0x2^7 + 0x2^8 + 0x2^9 + 0x2^{10} + 0x2^{11} = 5$



Changing multiple outputs

Finally, the Web-IO allows you to set multiple outputs at once while leaving others in their existing state.

Here you work with an output mask and the output value.

The output mask specifies which outputs are changed. The output value specifies which state these outputs should assume.

Example:
Output 1 should be switched ON,
Output 2 should be switched OFF,
Output 4 should be switched ON,
Output 5 should be switched OFF,
Output 10 should be switched ON,
Output 11 should be switched ON,

This means the following outputs are changed:

1,2,4,5,10 and 11.

The output mask is calculated as follows:

$0x2^0 + 1x2^1 + 1x2^2 + 0x2^3 + 1x2^4 + 1x2^5 + 0x2^6 + 0x2^7 + 0x2^8 + 0x2^9 + 1x2^{10} + 1x2^{11} = 3126$

In hexadecimal format this corresponds to 0xC36.

To calculate the output value, you only need to add the bit significances which correspond to the ON state, i.e. 1.

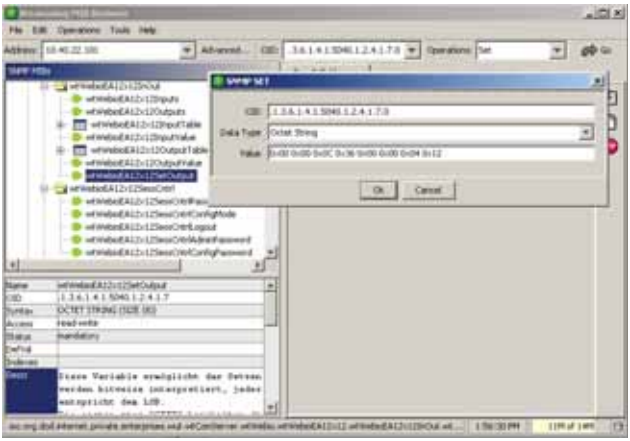
$1x2^1 + 1x2^4 + 1x2^{10} = 1042$

In hexadecimal format this corresponds to 0x412.

The output mask and value are sent using the SNMP variable

`wtWebioEA12x12SetOutput`

as an 8-byte octet string.



Changing the configuration

The basic settings of the Web-IO can likewise be read using SNMP and changed after a valid session login. The private MIB represents all the parameters which can be configured from the browser interface. In contrast to setting the outputs (takes

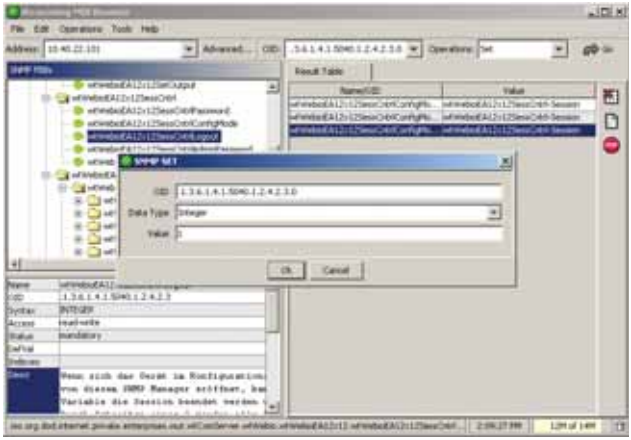
place immediately after a valid session is opened), the session must first be properly closed for the changes to take effect.

This is done by writing the variable

wtWebioEA12x12SessCntrlLogout

A value is used to determine how to quit the session:

1 All changes are saved
2 Quit without saving



If no SNMP communication takes place within 5 minutes during an open sesison, the Web-IO automatically quits the session and all changes are cancelled.

Opening an SNMP session has priority over a login. This means: A user with Config or Administrator rights loses his browser access as soon as an SNMP session is opened.

5.4.2 OPC - Standardized access

OPC (OLE for Process Control) is a software interface for accessing process data based on Microsoft's OLE technology.

Application programs such as visualization systems which use this interface are called OPC clients. On the other side of the interface you have OPC servers. These are device drivers which represent a particular hardware in abstract form as a set of OPC variables.

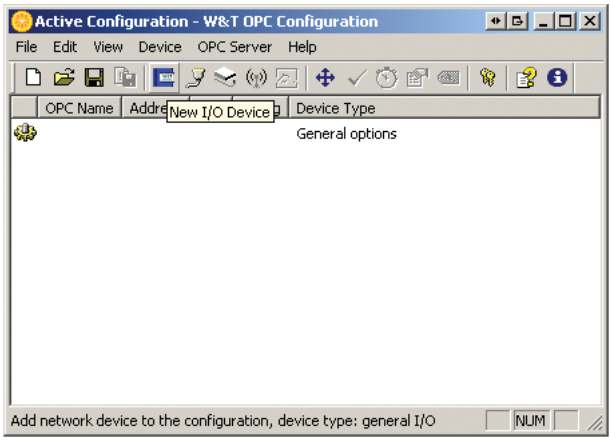
Installing the OPC server
You can find the OPC server on the product CD which accompanies the Web-IO in the Web-IO Digital section.

At our Web site www.WuT.de you will find at left the link „Search by article number." Enter here the article number of your Web-IO, select „Tools" from the field below and then click on „Go". On the resulting page select the link „OPC-Server".

For ease of downloading the required files are contained in a ZIP file. Save the extracted files in any desired directory on your hard drive (e.g. C:/Temp). Start the setup program for example by selecting *Run...* from the Start menu and then enter: „C\Temp\Setup". This installs and registers the OPC server on your computer.

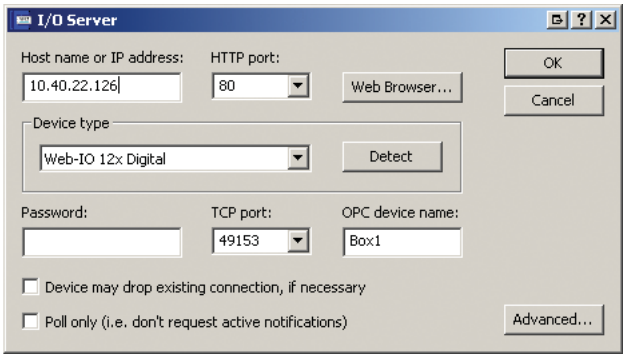
Uninstalling
The OPC server can be removed using the „Software" system component. You will find it there in the list as *W&T OPC-Server*

Configuration
First start the configuration program of the OPC server. For normal installations you will find the corresponding Start icon on your Windows interface under *Start >> Programs >> W&T OPC-Server Version 4 >> Configure OPC-Server.*


Active Configuration - W&T OPC Configuration
File Edit View Device OPC Server Help
OPC Name
New I/O Device
Device Type
General options
Add network device to the configuration, device type: general I/O
NUM

Click on the Web-IO icon or in the menu on *Device >> New I/O device.*

The following window opens:



Host name or IP address
Enter here the IP address of your Web-IO. If the address was also assigned a host name via DNA or a similar naming service, you may use this name instead of the IP address.

HTTP port
Enter here the port which the Web-IO will use as the HTTP port (factory default setting 80).

TCP port number
Must be the same as the Local Port entry in the Web-IO (factory default setting 491 53)

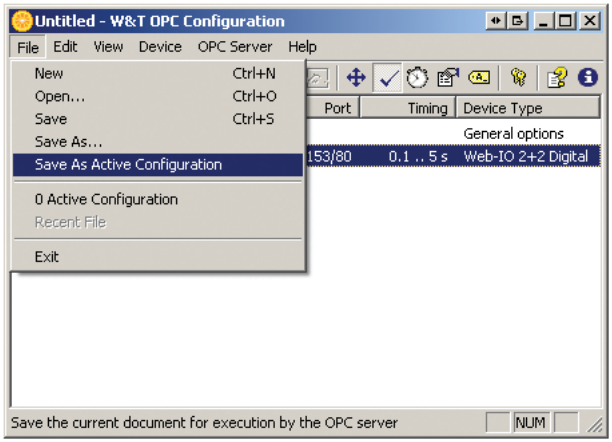
Password
Enter here the configurator or administrator password for the Web-IO. If you are uncertain of the correct password, please note after closing the dialog the status message for the device, in which any problems are indicated („Password required“ or „Incorrect password“).

Device type
Select here for example Web-IO 2xDigital

OPC device name
Enter here the name which you want the newly added device to be listed as under OPC in the front part of the OPC item name.

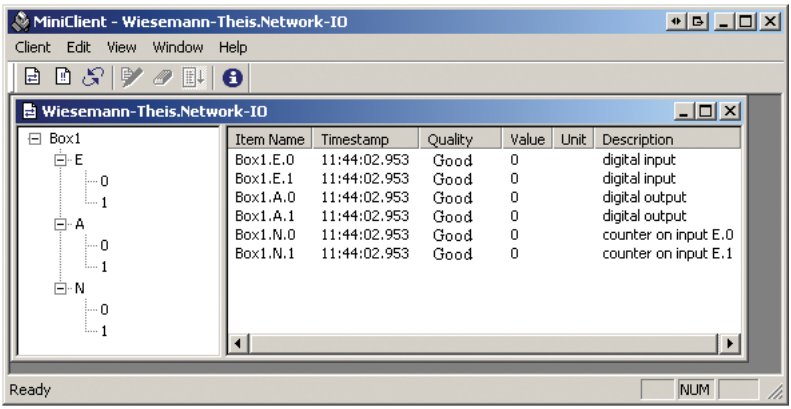
Poll only (do not subscribe to messages)
Skipping automatic messaging has the advantage that the amount of data traffic on the network does not depend on external events.

Finally you must save the configuration as an active configuration. Select *File >> Save as active configuration.*



OPC client applications can now access the Web-IO. The OPC server starts as a service and can be found in the OPC server list under the name *Wiesemann-Theis.Network-IO* .

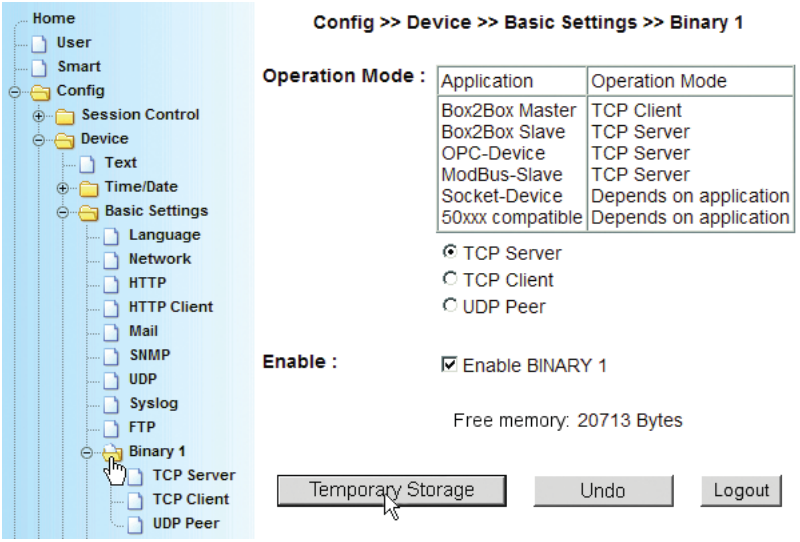
An overview of the items available in this name space is given by the OPC mini-client, which can be started under *Start >> Programs >> W&T OPC-Server Version 4 >> Minimal OPC-Client.*



Configuring the Web-IO as an OPC device

Required access rights: *Administrator*

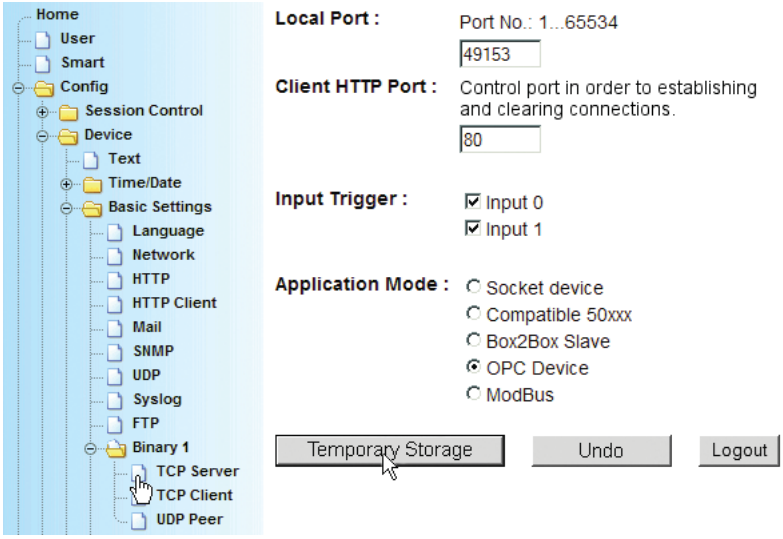
In the navigation tree of the Web-IO select *Config >> Device >> Basic Settings >> Binary 1*



For *Operation Mode* set *TCP-Server* mode.

Then click on the *Apply* button to send the changes to the Web-IO.

Now select in the navigation tree: *Config >> Device >> Basic Settings >> Binary 1 >> TCP-Server.*



Local Port
Unless your network administrator has indicated otherwise, you may use the factory default setting of 491 53 for the port.

Reasons for replacing the factory set local port could include for example a firewall which only allows certain port accesses.

In any case the set local port for the Web-IO must be identical to the corresponding settings in the OPC server.

Client HTTP port
Specifies the HTTP port on which the control connection to the OPC server should be established.

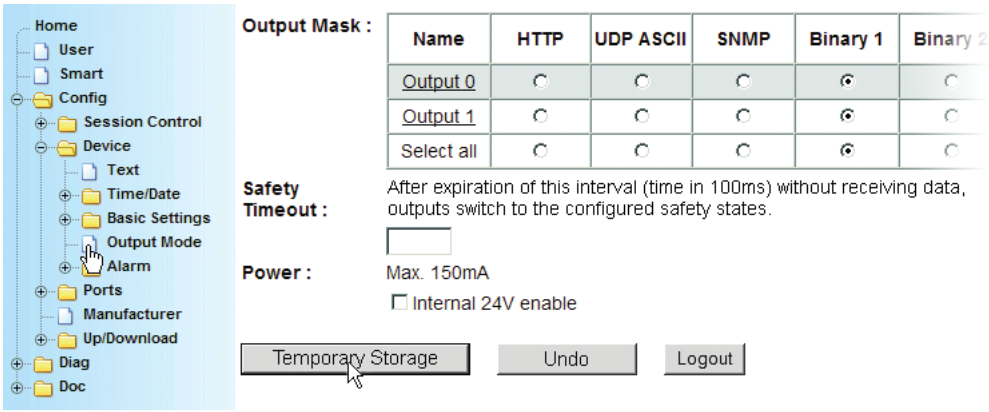
Unless otherwise specified, always use Port 80 here.

Input trigger
Here you activate the inputs which should trigger a message to the OPC server when there is a status change.

Application Mode
Select *OPC Device*

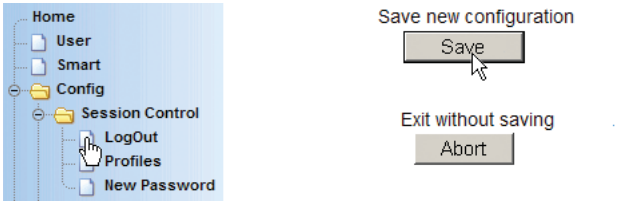
After all the parameters have been entered, confirm by clicking on the *Apply button*.

Now select in the navigation tree: *Config >> Device >> Output Mode*



Here you activate the outputs to be controlled by the OPC server and confirm by clicking on the *Apply* button.

Now you must activate the new settings. Use the *Logout* button or select *Config >> Session Control >> LogOut.*



After clicking on *Apply* all the settings are updated in the Web-IO and the Start page is reconstructed in Default User Mode. The Web-IO can now be accessed by the OPC server.

5.4.3 Modbus TCP - standardized access

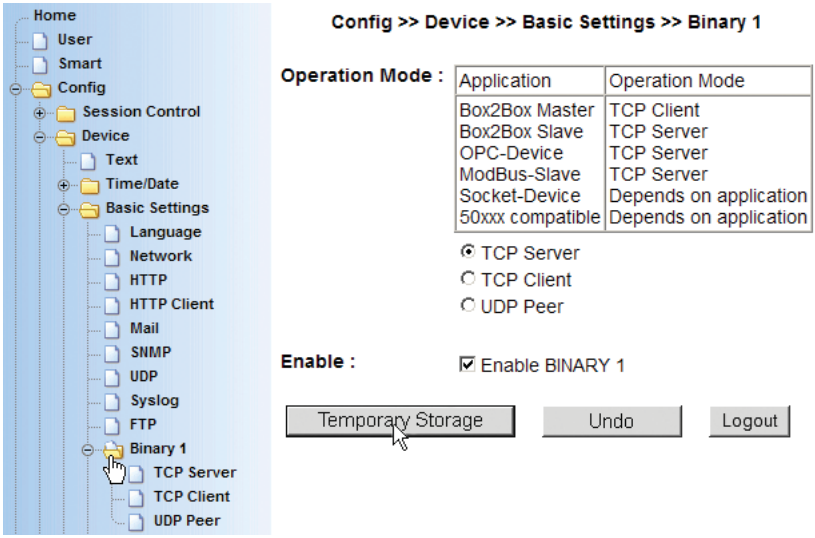
Modbus TCP is a software interface for address-based access to process data. The W&T Web-IOs when appropriately configured act as Modbus servers which can be controlled by Modbus controllers (clients or masters).

Using Modbus TCP access you can read the status of outputs, inputs and counters. Status and system polling is also possible. In addition the outputs can be switched via Modbus TCP and the counters cleared.

The Web-IO provides 64 16-bit registers which can be freely written and which can be accessed from the browser. Alarms can also be triggered by writing certain registers, which in turn results for example in the sending of e-mails or SNMP traps.

Configuration for Modbus TCP access

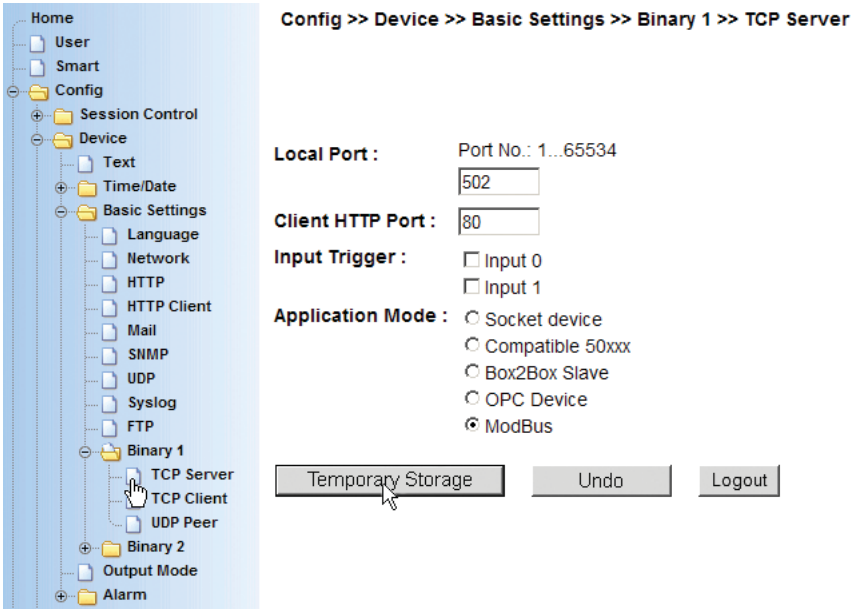
In the navigation tree of the slave Web-IO select *Config >> Device >> Basic Settings >> Binary 1*



For *Operation Mode* set *TCP-Server* and check *Enable Binary*.

Then click on the *Apply* button to send the changes to the Web-IO.

Now select in the navigation tree: *Config >> Device >> Basic Settings >> Binary 1 >> TCP-Server.*



Local Port
For use in normal Modbus TCP environments the local port of the Web-IO must be set to 502.

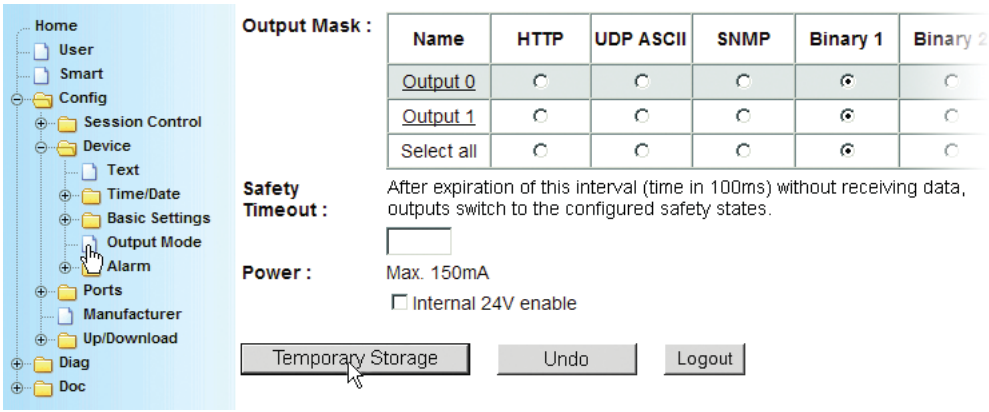
Client HTTP Port
This port is not used for Modbus TCP access and can remain at the factory default setting 80.

Input Triggers
The *Input Trigger* are not permitted to be enabled for Modbus TCP operation.

Application Mode
Check *Modbus*

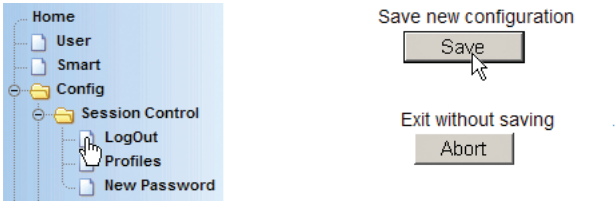
After all the parameters have been entered, confirm by clicking on the *Apply* button.

Now select in the navigation tree: *Config >> Device >> Output Mode*



Here you activate the outputs you want to be controlled by the Modbus client for Binary 1 and confirm by clicking on the *Apply* button.

Now you must still activate the new settings. Use the *Logout* button or select *Config >> Session Control >> Logout.*



After clicking on *Apply* all the settings are updated in the Web-IO and the Start page is reconstructed in Default User Mode. The Web-IO can now be accessed by the OPC client.

Modbus TCP communication

Modbus TCP is a master/slave procedure in which a Modbus master sends a request to the slave (Web-IO) and the slave (Web-IO) answers with a reply.

Data exchange between the Modbus client and the Web-IO takes place using Modbus TCP packets.

The Web-IO has a memory from which the Modbus master can read or to which it can write. This memory is divided into areas which represent certain characteristics of the Web-IO.

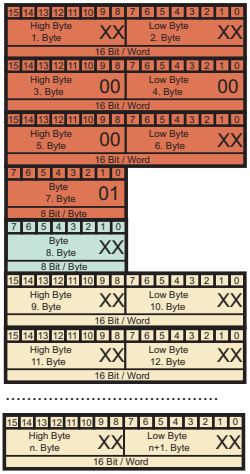
There are areas for the status of inputs, outputs and counters. Other areas indicate the system status or pertain to the alarms.

A special feature of the Web-IO is a memory area which on the one hand can be freely read or written via Modbus access and on the other hand can be called via the Web interface. With appropriate Web page programming a Modbus system can therefore be used for visualizing process data in the browser.

Another special function of the Web-IO is that an alarm can be triggered by writing to a particular memory area.

A list with the exact memory organization follows later in this section.

The Modbus data packets always consist of a header, the function code, the start address and other parameters and registers.


15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
High Byte 1. Byte XX
Low Byte 2. Byte XX
16 Bit / Word
High Byte 3. Byte 00
Low Byte 4. Byte 00
High Byte 5. Byte 00
Low Byte 6. Byte XX
7 6 5 4 3 2 1 0
Byte 7. Byte 01
8 Bit / Byte
Byte 8. Byte XX
High Byte 9. Byte XX
Low Byte 10. Byte XX
High Byte 11. Byte XX
Low Byte 12. Byte XX
High Byte n. Byte XX
Low Byte n+1. Byte XX

Transaction ID

Protocol ID

Length

Unit ID

Function Code

Start Address

Weitere Parameter oder Register

Transaction Identifier
Used to classify the reply from the Web-IO to the request of the client. The client normally increments the ID by 1 with each sending of data. The Web-IO always returns the received value 1:1.

Protocol Identifier
Has no meaning for communication with the Web-IO and is always 0x0000

Length
Number of bytes sent by *Length* (entire number of bytes sent - 6).

Unit identifier
Always 0x01 for Modbus TCP

Function Code
The *Function Code* specifies how the Modbus memory of the Web-IO is accessed:

- By bit
- By register (16-bits)

and what the purpose of the request is:

- 0x01 Read Coils - read individual bits
- 0x02 Read Discrete Inputs - read individual bits
- 0x03 Read Holding Registers- read multiple registers
- 0x04 Read Input Register - read multiple input registers.
- 0x05 Write Single Coil - write a bit
- 0x06 Write Single Register - write only one register
- 0x07 Read Exception State - read error status
- 0x0F Write Multiple Coils - write multiple bits
- 0x10 Write Multiple Register - write multiple registers

Start Address
Start Address specifies which area of the Modbus memory in the Web-IO will be accessed. This also determines which

characteristic is accessed (inputs, outputs, counters, alarms, ...).

A list with the exact memory organization can be found later in this section.

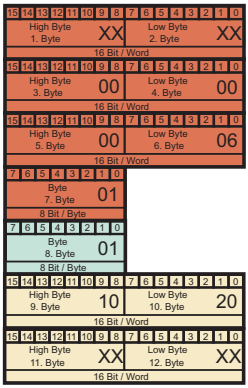
Both the Function Code *and* Start Address *determine what the sent request should trigger for the Web-IO.*

Other parameters and registers

Depending on which *Function Code* is used other parameters and registers may follow.

Function Code 0x01 Read Coils

Function Code 0x01 is intended for binary reading of the status of the Web-IO outputs.


15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
High Byte 1. Byte XX
Low Byte 2. Byte XX
16 Bit / Word
High Byte 3. Byte 00
Low Byte 4. Byte 00
16 Bit / Word
High Byte 5. Byte 00
Low Byte 6. Byte 06
16 Bit / Word
7 6 5 4 3 2 1 0
Byte 7. Byte 01
8 Bit / Byte
Byte 8. Byte 01
8 Bit / Byte
High Byte 9. Byte 10
Low Byte 10. Byte 20
16 Bit / Word
High Byte 11. Byte XX
Low Byte 12. Byte XX
16 Bit / Word

Transaction ID

Protocol ID

Length

Unit ID

Function Code

Start Address

Quantity of Outputs

Start Address

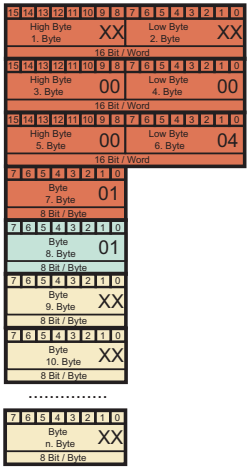
Specifies starting at which address bits (outputs) should be read.

Quantity of Outputs

Here the number of bits (outputs) to be read is transmitted.

By sending a data packet with FC 0x01 one or more Boolean values (output states, i.e. 0/1 or ON/OFF) can be requested.

The Web-IO replies as follows:



Bytes Count
Contains the number of send bytes which contain the requested bits. One byte per 8 started bits is sent. That means: starting with the 9th bit 2 bytes are sent.

Output State
Number of bytes, as sent in *Byte Count*. The first State Byte (10th byte) contains, beginning with Bit 0, the first 8 requested bits (outputs). If less then 8 bits were requested, the unused bytes are sent with 0. For more than 8 bits it continues with the 11th byte.

Example: Read the status of Outputs 0 and 1
Output 0 is OFF, Output 1 is ON

Start Address is set to 0x1020, *Quantity of Outputs* to 0x02. (The outputs can be called starting at address 0x1020.)

The Web-IO replies with *Byte Count* = 1, and *Output State* = 0x02.

Function Code 0x02 Read Discrete Inputs
Function Code 0x02 is intended for binary reading of the status of Web-IO inputs.

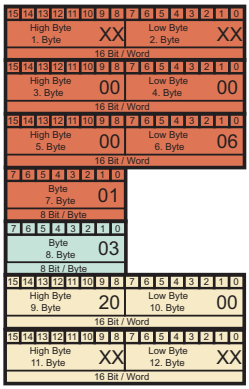
The packet construction and reply with *Function Code* 0x02 is identical to that of Function Code 0x01.

Example: Read the status of Inputs 0 and 1
Input 0 is OFF, Input1 is ON

Start Address is set to 1000, *Quantity of Inputs* to 2. (The inputs can be called starting at address 0x1000.)

The Web-IO replies with *Byte Count* = 1, and *Output State* = 0x02.

Function Code 0x03 Read Holding Registers
Function Code 0x03 is intended for reading multiple registers (16-bit values). Using FC 0x03 the values of inputs, outputs, counters etc. can be polled for the Web-IO depending on which *Start Address* is used.


15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
High Byte 1. Byte XX
Low Byte 2. Byte XX
16 Bit / Word
High Byte 3. Byte 00
Low Byte 4. Byte 00
16 Bit / Word
High Byte 5. Byte 00
Low Byte 6. Byte 06
16 Bit / Word
7 6 5 4 3 2 1 0
Byte 7. Byte 01
8 Bit / Byte
Byte 8. Byte 03
8 Bit / Byte
High Byte 9. Byte 20
Low Byte 10. Byte 00
16 Bit / Word
High Byte 11. Byte XX
Low Byte 12. Byte XX
16 Bit / Word

Transaction ID

Protocol ID

Length

Unit ID

Function Code

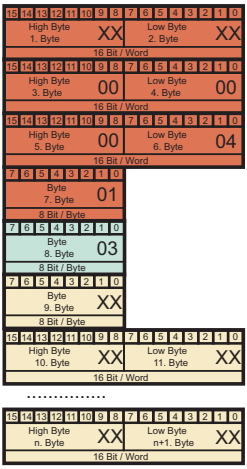
Start Address

Number of Registers

Start Address
Specifies the address to start reading the registers.

Number of Registers
Here the number of registers to read is transmitted.

The Web-IO replies with the following packet:



Bytes Count
Contains the number of bytes sent as registers (2 bytes per 16-bit register).

Register Value
One or more 16-bit register values. The first 16-bit register begins with the high byte at the position of the 10th byte.

Depending on the start address (beginning at 0x5000) two 16-bit values - i.e. 4 bytes - are passed for one requested 32-bit register. In this case again the value begins with the highest byte at the position of the 10th byte and the first low byte lies at the position of the 13th byte of the data packet.

Example:
Read the status of the outputs of a Web-IO 2xDigital
Output 0 is OFF, Output 1 is ON

Start Address is set to 0x2002,
Number of Registers to 0x01
(The outputs can be called as registers starting at address 0x2002.)

The Web-IO replies with

Byte Count = 0x02,

Register Value = 0x0002.

Function Code 0x04 Read Input Registers
Function Code 0x04 is especially provided for reading the status of the Web-IO inputs as 16-bit registers.

The packet structure of request and reply with Function Code 0x04 is identical to that of Function Code 0x03.

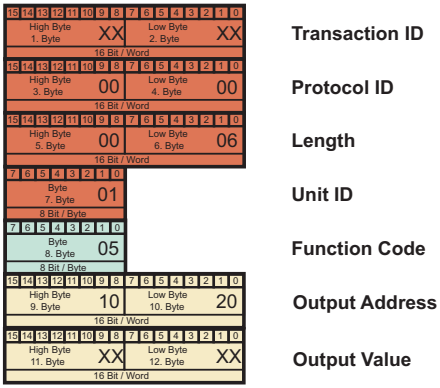
Example:
Read the status of the inputs of a Web-IO 2xDigital
Input 0 is OFF, Input1 is ON

Start Address is set to 0x2000,
Number of Registers to 0x01
(The inputs can be called as registers starting at address 0x2002.)

The Web-IO replies with

Byte Count = 0x02
Register Value = 0x0002

Function Code 0x05 Write Single Coil
Function Code 0x05 is intended for setting a single output.


15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
High Byte 1. Byte XX
Low Byte 2. Byte XX
16 Bit / Word
High Byte 3. Byte 00
Low Byte 4. Byte 00
High Byte 5. Byte 00
Low Byte 6. Byte 06
7 6 5 4 3 2 1 0
Byte 7. Byte 01
8 Bit / Byte
Byte 8. Byte 05
High Byte 9. Byte 10
Low Byte 10. Byte 20
High Byte 11. Byte XX
Low Byte 12. Byte XX
Transaction ID
Protocol ID
Length
Unit ID
Function Code
Output Address
Output Value

Output Address

Using *Output Address* you indirectly determine which output to switch. Beginning at 0x1020 each output has its own address.

Output Value

Here you specify whether an output should be turned ON or OFF:

ON = 0xFF00
OFF = 0x0000

The Web-IO replies with a data packet having the exact same structure.

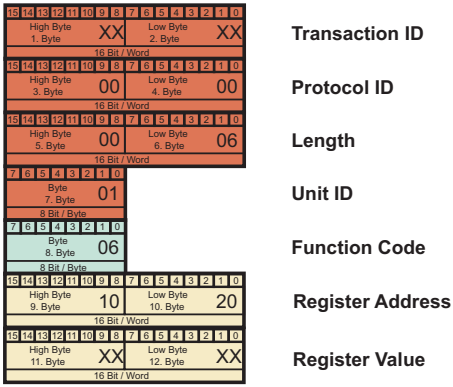
Example:
Output 1 of a Web-IO should be turned ON
Output Address is set to 0x1021,
Output Value to 0xFF00

The Web-IO replies with

Output Address = 1021
Output Value = 0xFF00

Function Code 0x06 Write Single Register

Function Code 0x06 works just the same way as Function Code 0x05 and is used for setting any desired register.

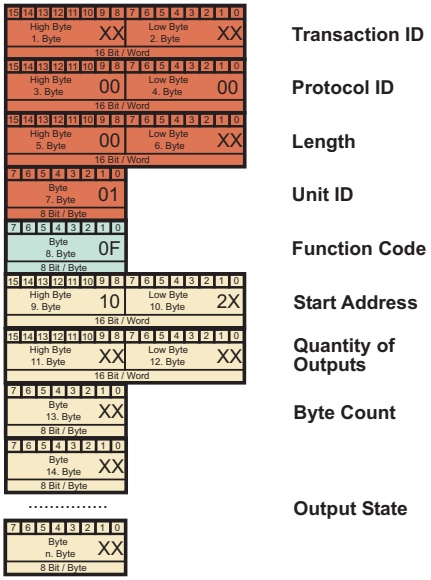

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
High Byte 1. Byte XX
Low Byte 2. Byte XX
16 Bit / Word
High Byte 3. Byte 00
Low Byte 4. Byte 00
16 Bit / Word
High Byte 5. Byte 00
Low Byte 6. Byte 06
16 Bit / Word
7 6 5 4 3 2 1 0
Byte 7. Byte 01
8 Bit / Byte
Byte 8. Byte 06
8 Bit / Byte
High Byte 9. Byte 10
Low Byte 10. Byte 20
16 Bit / Word
High Byte 11. Byte XX
Low Byte 12. Byte XX
16 Bit / Word
Transaction ID
Protocol ID
Length
Unit ID
Function Code
Register Address
Register Value

Register Address
Register Address specifies which register address to write to.

Register Value
A 16-bit register value which is written to the Modbus memory of the Web-IO.

The Web-IO replies with a data packet having the exact same structure.

Function Code 0x0F Write Multiple Coils
Function Code 0x0F is intended for bit setting of outputs.



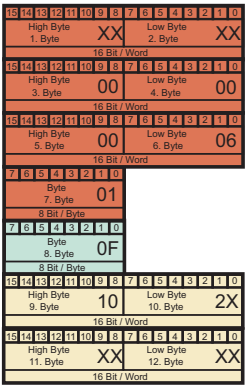
Start Address
Specifies at which address to begin writing output bits.

Quantity of Outputs
Here the number of bits to write is transmitted.

Bytes Count
Indicates the number of transmitted bytes which the output bits being set contain. One byte per 8 bits begun is sent. This means 2 bytes are sent beginning at the 9th bit.

Output State
Number of bytes as sent in *Byte Count*. The first byte (14th byte) contains, beginning at Bit 0, the first 8 output bits (outputs) to be set. If fewer than 8 bits (outputs) were specified, the unused bits are sent with 0. For more than 8 bits it continues with the 15th byte.

The Web-IO replies with a data packet having the following structure:



Transaction ID

Protocol ID

Length

Unit ID

Function Code

Start Address

Quantity of Outputs

Start Address
Specifies at which address to begin writing output bits.

Quantity of Outputs
Here the number of output bits written is sent.

Example:
Output 1 on a Web-IO should be turned ON
Output 0 should be turned OFF, Output 1 is ON

Output Address is set to 1020,
Quantity of Outputs to 0x02
Byte Count to 0x01

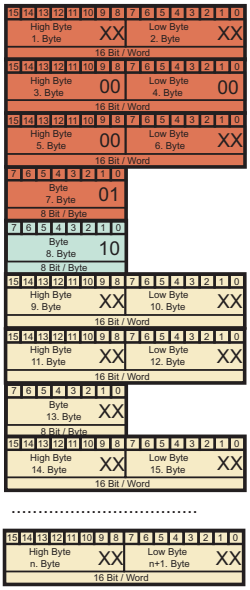
One byte Output Value with content 0x02 is sent.

The Web-IO replies with

Output Address = 1020
Quantity of Outputs = 0x0002

Function Code 0x10 Write Multiple Registers

Function Code 0x0F is intended for writing multiple 16-bit register values.


High Byte 1. Byte XX
Low Byte 2. Byte XX
16 Bit / Word
High Byte 3. Byte 00
Low Byte 4. Byte 00
16 Bit / Word
High Byte 5. Byte 00
Low Byte 6. Byte XX
16 Bit / Word
Byte 7. Byte 01
8 Bit / Byte
Byte 8. Byte 10
8 Bit / Byte
High Byte 9. Byte XX
Low Byte 10. Byte XX
16 Bit / Word
High Byte 11. Byte XX
Low Byte 12. Byte XX
16 Bit / Word
Byte 13. Byte XX
8 Bit / Byte
High Byte 14. Byte XX
Low Byte 15. Byte XX
16 Bit / Word
High Byte n. Byte XX
Low Byte n+1. Byte XX
16 Bit / Word

Transaction ID

Protocol ID

Length

Unit ID

Function Code

Start Address

Quantity of Registers

Byte Count

Register Value

Start Address

Specifies at which address to begin writing registers.

Quantity of Registers

Here the number of 16-bit registers to write is sent. When writing to the 32-bit area of the Web-IO two 16-bit registers must be counted per 32-bit value.

Bytes Count

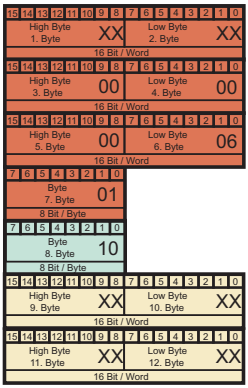
Contains the number of bytes to be sent. 2 bytes are counted for each 16-bit register to be sent.

Register Value

Here the 16-bit registers are sent. The first high byte is set as the 14th byte in the data packet, the first low byte as the 15th byte, etc.

When writing to the 32-bit area of the Web-IO (starting at address 0x5000) two 16-bit registers per 32-bit value beginning with the highest byte of the 32-bit value must be written.

The Web-IO replies with a data packet having the following structure:



Transaction ID

Protocol ID

Length

Unit ID

Function Code

Start Address

Quantity of Registers

Start Address

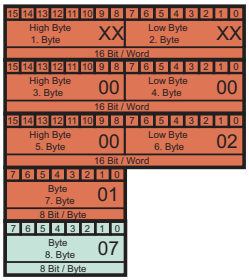
Specifies at which address to begin writing the 16-bit registers.

Quantity of Outputs

Here the number of written 16-bit registers is sent.

Function Code 0x07 Read Exception State

Function Code 0x07 is intended for reading the exception state (error status) of the Web-IO.


High Byte 1. Byte XX
Low Byte 2. Byte XX
16 Bit / Word
High Byte 3. Byte 00
Low Byte 4. Byte 00
16 Bit / Word
High Byte 5. Byte 00
Low Byte 6. Byte 02
16 Bit / Word
Byte 7. Byte 01
8 Bit / Byte
Byte 8. Byte 07
8 Bit / Byte

Transaction ID

Protocol ID

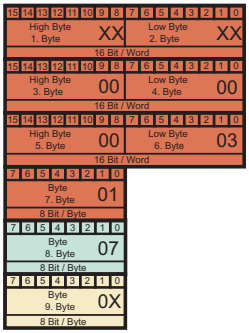
Length

Unit ID

Function Code

Packets sent by the master with *Function Code* 0x07 consist only of a Modbus header and *Function Code*. No other parameters are sent.

The Web-IO replies with a data packet having the following structure:


High Byte 1. Byte XX
Low Byte 2. Byte XX
16 Bit / Word
High Byte 3. Byte 00
Low Byte 4. Byte 00
16 Bit / Word
High Byte 5. Byte 00
Low Byte 6. Byte 03
16 Bit / Word
Byte 7. Byte 01
8 Bit / Byte
Byte 8. Byte 07
8 Bit / Byte
Byte 9. Byte 0X
8 Bit / Byte

Transaction ID

Protocol ID

Length

Unit ID

Function Code

Exception Code

Exception Code

For the Web-IO the Exception Code simply indicates the state of the LEDs for *System Error* and *On Error*.

Bit 0 = 1 corresponds to *System Error*
Bit 2 = 1 corresponds to *ON Error*

All other bits have no meaning for the Web-IO and are sent as 0.

Incorrect master requests

If the Modbus TCP master is supposed to send requests to the Web-IO which cannot be processed by the Web-IO in terms of form and number of parameters or with respect to the selected

register address, the Web-IO returns a Modbus packet whose structure corresponds to a reply to *Function Code* 0x07.

In such cases the Web-IO adds 0x80 to the received *Function Code* and sets this code into its own packet.

The *Exception Code* includes additional details about the error

0x01 Function Code not supported by Web-IO
0x02 Write attempt to read-only area
0x03 Write attempt with wrong data
0x04 Write attempt to allowed area failed

Modbus address areas for Web-IO Digital

All addresses are given in hex format.

The Web-IO has various Modbus memory areas:

- Bit area (beginning at address 0x1000),
- 16-bit area (beginning at address 0x2000)
- 32-bit area and 2x16-bit area (beginning at address 0x5000)

Addressing in the bit area is by the bit, i.e. 1 bit requires one address. In the 16-bit and 32-bit area addressing is by the word (2 bytes).

Inputs

Are located in the bit area beginning at address 0x1000
(Example 57637: 0x1000 and 0x1001),
in the 16-bit area beginning at 0x2000 in the 16-bit area beginning at 0x2000
(Example 57637: 0x2000),
in the 32-bit area beginning at 0x5000.

Outputs

Are located in the bit area beginning at address 0x1020
(Example 57637: 0x1020 and 0x1021)
in the 16-bit area beginning at 0x2002.
in the 32-bit area beginning at 0x5002.

Alarms
Are located in the bit area beginning at address 0x1040
(Example 57637: 0x1040 and 0x1041),
in the 16-bit area at 0x2004.
Alarm triggers are located in the bit area beginning at address 0x1800.

Counters
Are located in the 32-bit area beginning at address 0x5004
(Example 57637: 0x5004/0x5005 and 0x5006/0x5007).

Writing counter values: Counter can be set to any desired values.

Exception-Status
Located in the bit area beginning at address 0x1050,
in the 16-bit area at 0x200C (Low Byte).
Alternately the Exception Status is read using *Function Code* 0x07.

Configuration status
Located in the bit area at address 0x1058,
in the 16-bit area at 0x200C (High Byte).

Diagnostics status
(Number of errors) Lies in the 16-bit area at 0x2005,
in the 32-bit area at 0x5048.

Diagnostics status bits
Lie in the 16-bit area beginning at 0x2006,
in the 32-bit area beginning at 0x5048.
The Web-IO Digital allows 84 error messages.

A list of possible error messages can be found in the Appendix.

Device identification
Consists of a serial number (beginning at 0x6000) and Mac address (beginning at 0x6004).

Virtual registers for browse interaction

The memory area which the device provides for sending to Web applications, beginning at address 0x7000.

Reading non-supported registers

When reading data (memory areas) which were not defined for the device the unit returns „0“.

Modbus - virtual registers

The Web-IO provides 64 virtual 16-bit registers to which the Modbus master can write any desired values (High Byte first). Writing to these registers triggers no special actions in the Web-IO.

The virtual memory is used rather for sending Modbus process data to Web applications.

Using an HTTP Request

```
modbusreg?PW=<password>&
```

the 64 registers (128 bytes) can be called by Web applications.

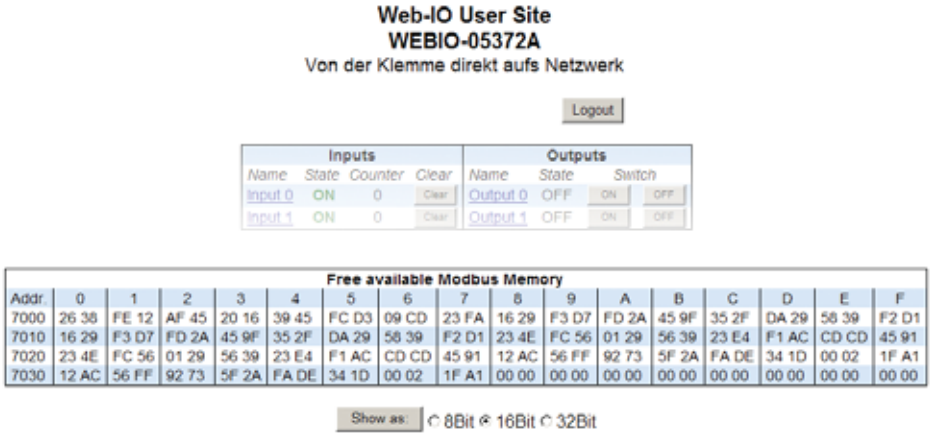
The Web-IO replies with

```
modbus;<HighByte1>;<LowByte1>;<HighByte2>;<LowByte2>;<HighByte3>;.....
```

In other words, all 64 registers (128 bytes) are output byte for byte separated by semicolons behind the word „modbus“.

Using JavaScript and programming techniques such as AJAX you can thereby implement process visualization in the browser.

In the simplest case the virtual registers can be displayed on the factory-side User page of the Web-IO (only if Modbus mode has been enabled in the Web-IO).


Web-IO User Site
WEBIO-05372A
Von der Klemme direkt aufs Netzwerk
Logout
Inputs
Name State Counter Clear
Input 0 ON 0 Clear
Input 1 ON 0 Clear
Outputs
Name State Switch
Output 0 OFF ON OFF
Output 1 OFF ON OFF
Free available Modbus Memory
Addr. 0 1 2 3 4 5 6 7 8 9 A B C D E F
7000 26 38 FE 12 AF 45 20 16 39 45 FC D3 09 CD 23 FA 16 29 F3 D7 FD 2A 45 9F 35 2F DA 29 58 39 F2 D1
7010 16 29 F3 D7 FD 2A 45 9F 35 2F DA 29 58 39 F2 D1 23 4E FC 56 01 29 56 39 23 E4 F1 AC CD CD 45 91
7020 23 4E FC 56 01 29 56 39 23 E4 F1 AC CD CD 45 91 12 AC 56 FF 92 73 5F 2A FA DE 34 1D 00 02 1F A1
7030 12 AC 56 FF 92 73 5F 2A FA DE 34 1D 00 02 1F A1 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Show as: 8Bit 16Bit 32Bit

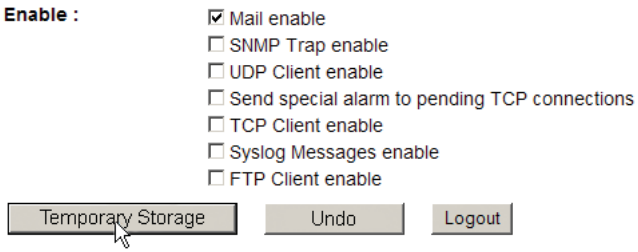
Modbus alarm triggering

Using an alarm trigger bit in the bit memory area you can trigger alarms. The bits begin at address 0x1800.

For models 57630, 57631 and 57634 there are 12 alarms available. Model 57637 provides 4 alarms.

In the browser configuration interface for example under *Config >> Device >> Alarm >> Alarm1* you select how the alarm should be sent. Here for example via e-mail.

Required access rights: *Administrator*

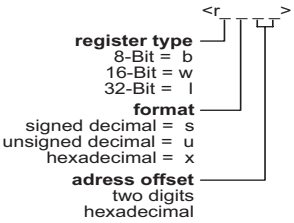


For an alarm triggered by Modbus there should be no other triggers (*Input Trigger*, *Output Trigger*, *System Trigger*) activated.

The remaining configuration of the alarms is done normally as described in the following section „E-mail and more - alarm when status changes“.

Using special tags when sending alarms (mail, FTP, SNMP) you can also embed a representation of the virtual memory into the alarm message. The tag stands so to speak as a placeholder for the contents of the selected register.

The tags allow representation in 8-, 16- or 32-bit manner. You can specify whether the addressed value is output in decimal or hexadecimal. In addition the memory offset is given starting at address 0x7000.



Example: At address 7004 Modbus wrote the 16-bit value 0x4711.

The alarm text was configured in the Web-IO as follows:

„<t> - Register 4 has the following value: <rwx04>“

The received alarm message would look like this:

„2011-02-22 10:34:21 - Register 4 has the following value: 4711“

All the details and possibilities for Web-IO alarm messages are described in the following section „E-mail and more - alarm when status changes“.

Modbus memory organization

-bit area:

addresse (hexadec.)	description	memory type	length (byte)	read bits with FC	read reg. with FC	Write bits with FC	write reg. with FC
1000	Input 0	bit	1	0x01, 0x02	-	-	-
1001	Input 1	bit	1	0x01, 0x02	-	-	-
1002	Input 2	bit	1	0x01, 0x02	-	-	-
1003	Input 3	bit	1	0x01, 0x02	-	-	-
1004	Input 4	bit	1	0x01, 0x02	-	-	-
1005	Input 5	bit	1	0x01, 0x02	-	-	-
1006	Input 6	bit	1	0x01, 0x02	-	-	-
1007	Input 7	bit	1	0x01, 0x02	-	-	-
1008	Input 8	bit	1	0x01, 0x02	-	-	-
1009	Input 9	bit	1	0x01, 0x02	-	-	-
100A	Input1 0	bit	1	0x01, 0x02	-	-	-
100B	Input 11	bit	1	0x01, 0x02	-	-	-
1020	Output 0	bit	1	0x01, 0x02	-	0x05	0x0F
1021	Output 1	bit	1	0x01, 0x02	-	0x05	0x0F
1022	Output 2	bit	1	0x01, 0x02	-	0x05	0x0F
1023	Output 3	bit	1	0x01, 0x02	-	0x05	0x0F
1024	Output 4	bit	1	0x01, 0x02	-	0x05	0x0F
1025	Output 5	bit	1	0x01, 0x02	-	0x05	0x0F
1026	Output 6	bit	1	0x01, 0x02	-	0x05	0x0F
1027	Output 7	bit	1	0x01, 0x02	-	0x05	0x0F
1028	Output 8	bit	1	0x01, 0x02	-	0x05	0x0F
1029	Output 9	bit	1	0x01, 0x02	-	0x05	0x0F
102A	Output 10	bit	1	0x01, 0x02	-	0x05	0x0F
102B	Output 11	bit	1	0x01, 0x02	-	0x05	0x0F
1040	Alarm state 1	bit	1	0x01, 0x02	-	-	-
1041	Alarm state 2	bit	1	0x01, 0x02	-	-	-
1042	Alarm state 3	bit	1	0x01, 0x02	-	-	-
1043	Alarm state 4	bit	1	0x01, 0x02	-	-	-
1044	Alarm state 5	bit	1	0x01, 0x02	-	-	-
1045	Alarm state 6	bit	1	0x01, 0x02	-	-	-
1046	Alarm state 7	bit	1	0x01, 0x02	-	-	-
1047	Alarm state 8	bit	1	0x01, 0x02	-	-	-
1048	Alarm state 9	bit	1	0x01, 0x02	-	-	-
1049	Alarm state 10	bit	1	0x01, 0x02	-	-	-
104A	Alarm state 11	bit	1	0x01, 0x02	-	-	-
104B	Alarm state 12	bit	1	0x01, 0x02	-	-	-
1060	Exception State	bit	1	0x01, 0x02	-	-	-
1068	Config. state	bit	1	0x01, 0x02	-	-	-
1800	Alarm trigger 1	bit	1	0x01, 0x02	-	0x05	0x0F
1801	Alarm trigger 2	bit	1	0x01, 0x02	-	0x05	0x0F
1802	Alarm trigger 3	bit	1	0x01, 0x02	-	0x05	0x0F
1803	Alarm trigger 4	bit	1	0x01, 0x02	-	0x05	0x0F
1804	Alarm trigger 5	bit	1	0x01, 0x02	-	0x05	0x0F
1805	Alarm trigger 6	bit	1	0x01, 0x02	-	0x05	0x0F
1806	Alarm trigger 7	bit	1	0x01, 0x02	-	0x05	0x0F
1807	Alarm trigger 8	bit	1	0x01, 0x02	-	0x05	0x0F
1808	Alarm trigger 9	bit	1	0x01, 0x02	-	0x05	0x0F
1809	Alarm trigger 10	bit	1	0x01, 0x02	-	0x05	0x0F
180A	Alarm trigger 11	bit	1	0x01, 0x02	-	0x05	0x0F
180B	Alarm trigger 12	bit	1	0x01, 0x02	-	0x05	0x0F

Please note that depending on which Web-IO model are you using the entire width of the inputs, outputs, counters or alarms may not be available.

16- and 32-bit area:

addresse (hexadec.)	description	memory type	length (byte)	read bits with FC	read reg. with FC	Write bits with FC	write reg. with FC
2000	Inputs 0 - 11	16-bit	2	-	0x03, 0x04	-	-
2002	Outputs 0 - 11	16-bit	2	-	0x03, 0x04	-	-
2004	Alarm state 1 - 12	16-bit	2	-	0x03, 0x04	-	-
2006	Diagnosis Error count	16-bit	2	-	0x03, 0x04	-	0x06, 0x10
2007	Diagnostic state 0 - 15	16-bit	2	-	0x03, 0x04	-	-
2008	Diagnostic state 16 - 31	16-bit	2	-	0x03, 0x04	-	-
2009	Diagnostic state 32 - 47	16-bit	2	-	0x03, 0x04	-	-
200A	Diagnostic state 48 - 63	16-bit	2	-	0x03, 0x04	-	-
200B	Diagnostic state 64 - 79	16-bit	2	-	0x03, 0x04	-	-
200C	Diagnostic state 80 - 95	16-bit	2	-	0x03, 0x04	-	-
200D	Exception/Conf.-State	16-bit	2	-	0x03, 0x04	-	-
5000	Inputs 0 - 11	32-bit	4	-	0x03, 0x04	-	-
5002	Outputs 0 - 11	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
5004	Alarm state 1 - 12	32-bit	4	-	0x03, 0x04	-	
5006	Counter 0	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
5008	Counter 1	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
500A	Counter 2	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
500C	Counter 3	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
500E	Counter 4	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
5010	Counter 5	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
5012	Counter 6	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
5014	Counter 7	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
5016	Counter 8	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
5018	Counter 9	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
501A	Counter 10	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
501C	Counter 11	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
504A	Diagnosis Error count	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
504C	Diagnostic state 0 - 31	32-bit	4	-	0x03, 0x04	-	-
504E	Diagnostic state 32 - 63	32-bit	4	-	0x03, 0x04	-	-
5050	Diagnostic state 64 - 95	32-bit	4	-	0x03, 0x04	-	-
7000	virtuel Register 0	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
7002	virtuel Register 1	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
7004	virtuel Register 2	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
7006	virtuel Register 3	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
7008	virtuel Register 4	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
700A	virtuel Register 5	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
700C	virtuel Register 6	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
700E	virtuel Register 7	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
7010	virtuel Register 8	32-bit	4	-	0x03, 0x04		0x06, 0x10
.....	virtuel Register 9 - 23	32-bit	4				
702E	virtuel Register 23	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
7030	virtuel Register 24	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
7032	virtuel Register 25	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
7034	virtuel Register 26	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
7036	virtuel Register 27	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
7038	virtuel Register 28	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
703A	virtuel Register 29	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
703C	virtuel Register 30	32-bit	4	-	0x03, 0x04	-	0x06, 0x10
703E	virtuel Register 31	32-bit	4	-	0x03, 0x04		0x06, 0x10

Please note that depending on which Web-IO model are you using the entire width of the inputs, outputs, counters or alarms may not be available.

5.5 Email and more - Alarms

In the Web-IO you can specify up to 12 different alarm conditions. Alarm messages can be issued depending on the status of the inputs and outputs. Four different network protocols are available:

- Mail (SMTP)
- SNMP
- Syslog
- UDP
- TCP (on an existing connection)
- TCP Client
- FTP Client

In addition to the alarm messages, the Web-IO also sends Alarm Clear messages when the alarm conditions are no longer met, i.e. the normal state is present again.

5.5.1 Determining the alarm conditions

Required access rights: *Administrator*

The alarm conditions can be specified in various trigger groups.

- Input Trigger
- Output Trigger
- System Trigger

The conditions for different trigger groups should not be mixed within an alarm.

Make triggering of an alarm dependent either on the status of the inputs or by one of the system triggers!

A distinction is made between two basic types of alarm triggers:

- Trigger with alarm state and normal state
- Trigger as single event

Trigger with alarm state and normal state
The trigger condition may be states ON or OFF for the in- or outputs, but may also be the system trigger Load Control.

If for example ON is selected as a trigger for an input, the alarm message is sent when this input goes to ON. When the input returns to OFF, the Web-IO sends the Alarm Clear message.

If an alarm condition returns to the normal state before the Web-IO was able to send the alarm message, only the Alarm Clear message is sent.

You could therefore say that these alarms are always finished with an Alarm Clear message.

Trigger as single event
Here triggers such as cold and warm start, timer interval and the ANY setting count for the inputs and outputs.

There is no normal state as such for these one-time events. SInce alarms shoudl always be finished with an Alarm Clear message, this type of trigger always generates only one Alarm Clear message.

There is no alarm message, but rather only Alarm Clear messages.

Alarm based on the status of the inputs and outputs
Select in the navigation tree *Config >> Device >> Alarm >> Alarm1* if for example you want to process the alarm conditions for Alarm 1.

Under *Input Trigger* and *Output Trigger* you can specify the following:

1. Which inputs or outputs to check for status
2. Which state (ON or OFF) meets the condition (AND operation)

The trigger conditions of the blocks Input Trigger, Output Trigger, Interval Timer and Load Control can be set independently (OR operation)

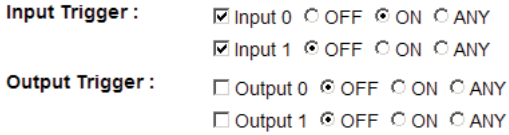
All inputs/outputs which are selected must meet the specified condition for the alarm to be triggered (AND operation).

If multiple inputs/outputs are configured with the ANY condition, these must switch at the same time for the alarm to be triggered.

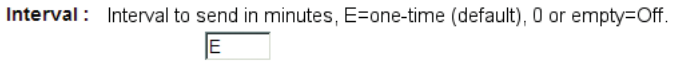
Alarms triggered using ANY only generate one Alarm Clear message!

Example:
Trigger an alarm when Input 0 and Input 1 are ON and no signal (=OFF) is on Input 1. Inputs 3 -11 are ignored.

The configuration would then look like this:



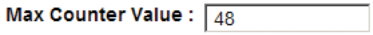
In addition you must specify whether an alarm is triggered only once when the alarm condition is met. Alternately you can specify an interval at which the alarm is repeated as long as the alarm condition persists.



In the Interval field you can specify either *E* for one time or an interval in minutes. Theoretical values between 1 and 4.2 billion minutes are possible. If the field is blank or 0, no alarm will be issued.

Alarm when a counter value is reached

Instead of responding to the state of the inputs and outputs, you can trigger an alarm when a counter state is reached. To do this you must enter a value between 1 and 4.2 billion in the Max Counter Value field. An alarm is issued when one of the activated inputs reaches this counter value.

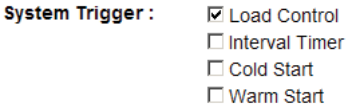


Counter and input patterns are mutually exclusive as alarm conditions. As soon as you have entered a value in Max Counter Value, the state of the inputs is no longer used as an alarm condition, and only the counter state is monitored.

Alarm for load error

The outputs of the Web-IO Digital have thermal overload monitoring. If an overload is detected on an output, it is immediately turned off. If no load is detected it works also.

Over- or underload can also be used as an alarm condition. To do this, select *Load Control* under *Config >> Device >> Alarm >> Alarm1, System Trigger.*



When an overload occurs, the corresponding output can be enabled again after eliminating the overload by opening Diag >> Report in the navigation tree and clicking on the Cancel Report button.

An Alarm Clear message is issued when the affected output is enabled.

Time-based alarm

Activating Interval Timer can allow the Web-IO to regularly issue alarms at intervals specified under *Interval* regardless of the input and output state.

System Trigger :

- ☐ Load Control
- ☑ Interval Timer
- ☐ Cold Start
- ☐ Warm Start

In this way you can for example implement a keep-alive monitor.

At which times the alarm is triggered is determined in the area *Time Trigger*.

Time Trigger : Uhrzeitgesteuerte Ausgabe des Alarms

Feld	Eingabe [Zahl * , -]	möglicher Zahlenbereich
Minute	0-59/15	0-59
Stunde	8-17	0-23 (0 ist Mitternacht)
Monatstag	1-31	1-31
Monat	1-12	1-12
Wochentag	1-5	0-6 (0 ist Sonntag)

The syntax for the time setting is derived from the Cron format used in Linux/Unix. Multiple entries separated by commas can be entered in one field.

The „*“ stands for any whole-number value. Example: * In the Minute field there is an alarm every minute, even if the conditions in the other fields are met.

Appending a „/“ to the actual value or value range allows you to specify a cycle value behind it. Example: 0-59/5 in the Minute field means there is an alarm every five minutes, even if the conditions in the other fields are met.

If you want to work with cyclical alarms, no wildcards „“ are allowed in front of the „/“.*

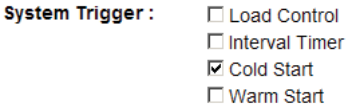
Using „-“ allows you to use time spans as triggers. Example: 8-17 in the Hour field means every hour between 8:00 a.m. and 5:00 p.m..

The above fields then mean an alarm is triggered every month on every Monday through Friday between 8:00 a.m. and 5:00 p.m. every five minutes.

It makes no sense to enter anything other than „E“ in the Interval *input field if you are not using* Time Trigger*!*

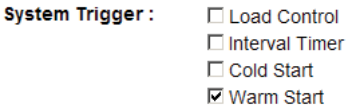
For timer-controlled alarms, the alarm is sent in the form of the Alarm Clear message

Alarm at cold start
Activating cold start causes the Web-IO to trigger an alarm after a Hardware or Power-ON Reset.



For cost start-controlled alarms, the alarm is sent in the form of the Alarm Clear message

Alarm at warm start
Activating warm start causes the Web-IO to send an alarm after a software reset, e.g., Connect to TCP Port 8888..



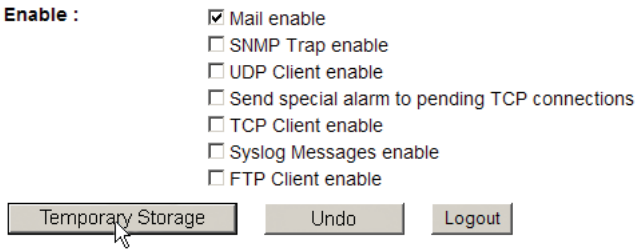
For warm start-controlled alarms, the alarm is sent in the form of the Alarm Clear message

After the alarm conditions have been specified, you can select which path to use for issuing the alarm.

5.5.2 E-Mail-Alarm - SMTP

To send an alarm via e-mail you must also activate *Mail Enable* (in addition to the alarm conditions) under *Config >> Device >> Alarm >> Alarm1*.

Required access rights: *Administrator*



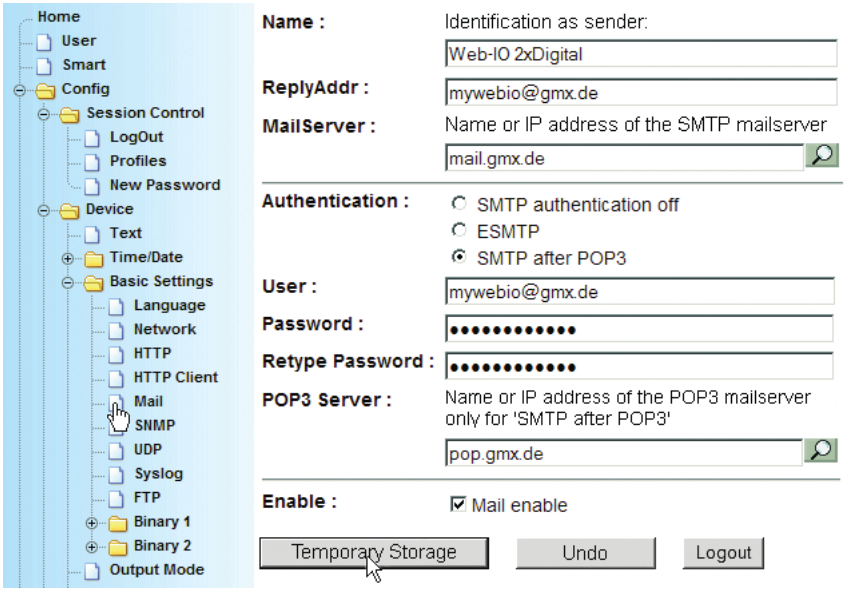
Send these settings to the Web-IO by clicking on the *Temporary Storage* button.

Basic settings for sending by e-mail

Required access rights: *Administrator*

In order for the Web-IO to send e-mail in the first place, you must make a few basic settings.

Go to *Config >> Device >> MailAdmin:*



Fill in the fields according to your application:

Name
Name entered as the sender when sending the e-mail.

Reply Address
E-mail address entered as the destination address in the outgoing e-mail when the recipient of an alarm e-mail selects *Reply*. SInce the Web-IO itself can only send but not receive e-mail, you should enter here the e-mail address of the person who has technical responsibility for the Web-IO.

Mail Server
Here you must enter the host name or the IP address of the SMTP server used to send its alarm mails through the Web-IO.

Authentification
Here you can select whether you want to use e-mail authentication. depending on the ISP requirements, the Web-IO can be configured for a login with ESMTP or SMTP after POP3.

User
Enter here the user name for the e-mail account.

Password / Retype Password
Here the password assigned for the e-mail account must be entered.

POP3 Server
If you are using SMTP after POP3, you must enter here the name or the IP address of the POP3 server.

Enable
Set Mail enable to activate sending of e-mail.

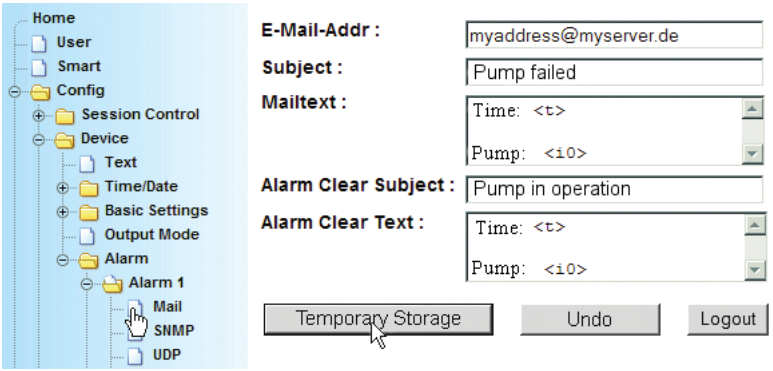
If the name of the SMTP or POP3 server is specified, a DNS server must be specified in the network settings.

Send the settings to the Web-IO by clicking on the *Temporary Storage* button.

Specifying the e-mail recipient, subject and text

Required access rights: *Administrator*

Under *Config >> Device >> Alarm >> Alarm1 >> Mail* you can now specify who should receive an alarm mail, what should appear as the subject and what text the e-mail contains.



E-Mail Addr.
Enter the e-mail address for sending the e-mail when the alarm conditions are met. If you want multiple recipients for the e-mail, you must separate the e-mail addresses with colons, e.g. user1@domain.de; user2@domain.de; ...

Subject
Enter here the text to appear in the subject line of the e-mail.

Mailtext
This can be a variable text. In addition you can use alarm variables. These are placeholders for the state of the individual inputs and outputs as well as for the counter states.

Alarm Clear Subject
As soon as the configured alarm condition is no longer present, an Alarm Clear message is sent with the subject line entered here.

Alarm Clear Text

As with the mail text, here you may enter a freely formulated text for the Alarm Clear message.

In addition, alarm variables can be used in the subject line and in the mail text. These are placeholders for status and name of of the individual inputs and outputs, as well as for the counter states of the counters. In addition, placeholders can be used for Date and Device Name.

Alarm Variable	Description
<dn>	Device Name look at: Config>>Device>>Text)
<i>	Input state as bit pattern in hex format
<i**x**>	State of Input No. **x** (ON/OFF),
<in**x**>	Name of Input No. **x**
<o>	Output state as bit pattern in hex format
<o**x**>	State of Output No. **x** (ON/OFF),
<on**x**>	Name of Output No. **x**
<c**x**>	Count state of counter no. **x** in decimal format
<t>	Time stamp with date and time formatted DD.MMM.YYYY hh:mm:ss
<$y>	Year formatted "YYYY"
<$m>	Month formatted "MM"
<$d>	Day formatted "DD"
<$h>	Hour formatted "hh"
<$i>	Minutes formatted "mm"
<$s>	Seconds formatted "ss"
	depending on the Web-IO Type **x** can be between 0 and 23

The example entered in the configuration frame would then look as follows as an e-mail:

Subject: tank 1 is empty

time: 18.Nov.2003 13:44:38

pump 1 = OFF

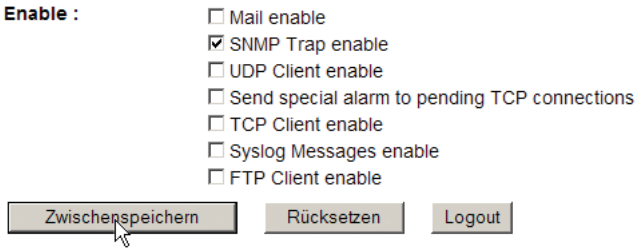
The mail tags were replaced by the values current at the time the mail was sent.

5.5.3 SNMP Alarm

In networks with a network management program you can also use SNMP protocol for sending alarms.

Under *Config >> Device >> Alarm >> Alarm1* you must also activate *SNMP Trap Enable* in addition to the alarm settings.

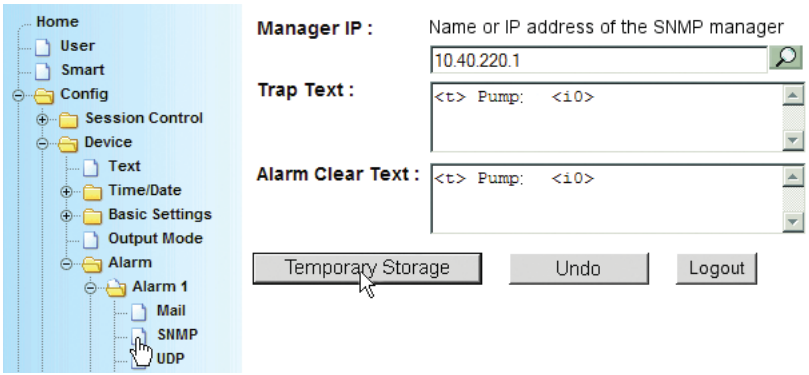
Required access rights: *Administrator*



Send the settings to the Web-IO by clicking on the *Temporary Storage* button.

Select in the navigation tree *Config >> Device >> Alarm >> Alarm1 >> SNMP*

Required access rights: *Administrator*



Manager IP
Enter here the IP address of the network management system that will receive the SNMP information.

Trap-Text
Just as with the mail function, you can also enter a „trap text“ here which is sent to the manager software.

To incorporate the status of inputs and outputs as well as counter states into the text, you can likewise use alarm variables.

Alarm Clear Text
As in the case of the trap text, here you may enter here a freely formulated text for the Alarm Clear message.

The maximum length of the trap text must not exceed 450 bytes. Longer texts are cut off by the Web-IO.

Alarm Variable	Description
<dn>	Device Name look at: Config>>Device>>Text)
<i>	Input state as bit pattern in hex format
<i**x**>	State of Input No. **x** (ON/OFF),
<in**x**>	Name of Input No. **x**
<o>	Output state as bit pattern in hex format
<o**x**>	State of Output No. **x** (ON/OFF),
<on**x**>	Name of Output No. **x**
<c**x**>	Count state of counter no. **x** in decimal format
<t>	Time stamp with date and time formatted DD.MMM.YYYY hh:mm:ss
<$y>	Year formatted "YYYY"
<$m>	Month formatted "MM"
<$d>	Day formatted "DD"
<$h>	Hour formatted "hh"
<$i>	Minutes formatted "mm"
<$s>	Seconds formatted "ss"
	depending on the Web-IO Type **x** can be between 0 and 23

If you have entered an Administrator password, you will ave to enter this in your manager software as a „community string“!

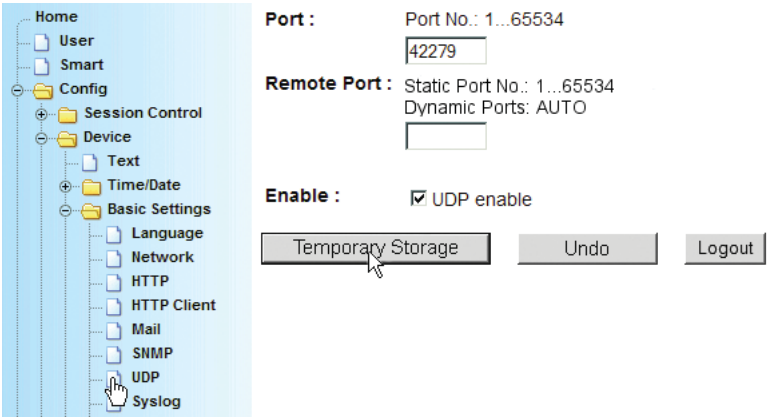
5.5.4 Alarm via UDP

Another way to pass an alarm to another network station is provided by UDP protocol. Here a corresponding program must be running on those PCs that are supposed to receive such an alarm.

To enable UDP for alarm sending, select *Config >> Basic Settings >> UDP* in the navigation tree.

Required access rights: *Administrator*

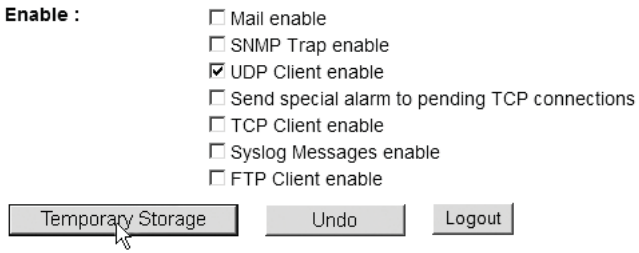
In the Port field enter the desired local port. The factory default setting is Port 42279. Check the UDP-enable box.



Send the selected settings to the Web-IO by clicking on the *Save* button.

To activate UDP for alarm transmission, select in the navigation tree *Config >> Device >> Alarm >> Alarm1* and select *UDP Client Enable* in the *Enable* area.

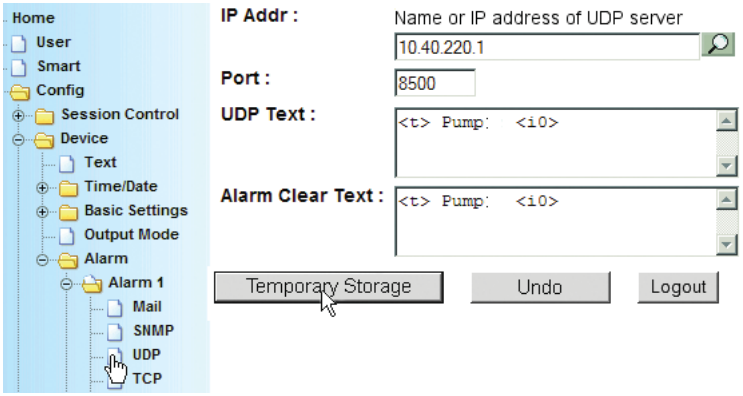
Required access rights: *Administrator*



Send the settings to the Web-IO by clicking on the *Temporary Storage* button.

Under *Config >> Device >> Alarm >> Alarm1 >> UDP* the UDP-specific configurations can then be entered.

Required access rights: *Administrator*



IP Addr

Here you enter the IP address of the network station you want to receive the alarm message. Alternately you can enter a broadcast IP address to make the alarm messages accessible to all the participants in the network segment. IP-Adresse 172.16.232.255 for example would allow all network participants in segment 172.16.232.0 to receive the alarm.

Port

In this field you enter the local UDP port on which the application program should receive the UDP alarm messages.

UDP Text

Just as with the mail and SNMP function, you can enter here a text which is sent to the UDP application. To incorporate the state of inputs and outputs or counter states into the text, you can also use the e-mail tags.

Send the settings to the Web-IO by clicking on the *Temporary Storage* button.

Switching outputs for a different Web-IO using UDP

Sending of UDP datagrams can also be used for sending commands to switch outputs on another Web-IO. To do this the address of the other Web-IO must be entered as the *IP Addr* and local port of the other Web-IO as the local port.

„localhost“ or „127.0.0.1“ or the Web-IO‘s own address may be entered as the address. The Web-IO then sends the command to itself, which enables switching of its own outputs using the alarms.

Under UDP Text the corresponding command for switching an output is entered.

e.g.

```
GET/outputaccess0?PW=<password>&State=ON&NA=ON&
```

for turning on Output 0 or.

```
GET/outputaccess0?PW=<password>&State=OFF&NA=ON&
```

for turning it off.

A linefeed should be added to the command by pressing the Enter key.

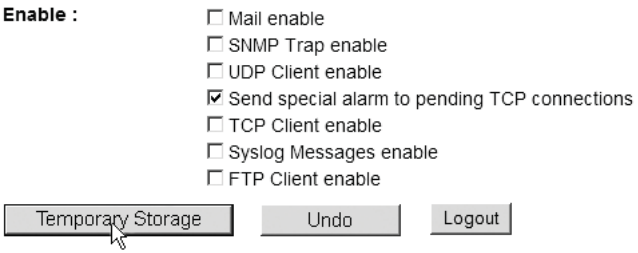
Details on the commands can be found in section 5.6.2

Please note that the trigger for the alarm must be present for min. 1 second for the command to be reliably sent.

5.5.5 Alarm message to your own TCP applications

In applications which communicate with the Web-IO over the socket interface via TCP, existing TCP connections can be used to send the input state when there is a state change.

So that a TCP data packet is sent to the active TCP client application when the alarm condition is met, go to *Config >> Device >> Alarm >> Alarm1* in the *Enable* area and select *Send input to pending TCP connections*



Send the settings to the Web-IO by clicking on the *Temporary Storage* button.

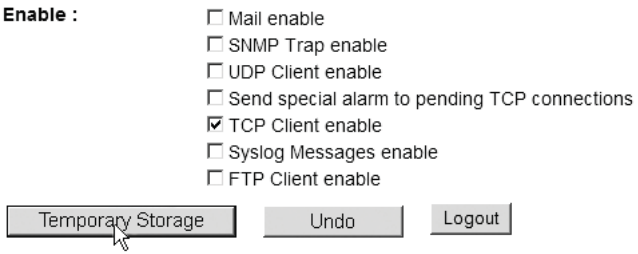
Once you have made all the entries pertaining to the alarm, activate the new settings using *Config >> Session Control >> LogOut, Save.*

5.5.6 Alarm message to a TCP server

You can also send an alarm messag to a TCP server. If there is an alarm the Web-IO opens a TCP connection to the configured Server and sends the message. The server hat to close the connection after recept of message.

To activate TCP client for alarm transmission, select in the navigation tree *Config >> Device >> Alarm >> Alarm1* and select *TCP Client Enable* in the *Enable* area.

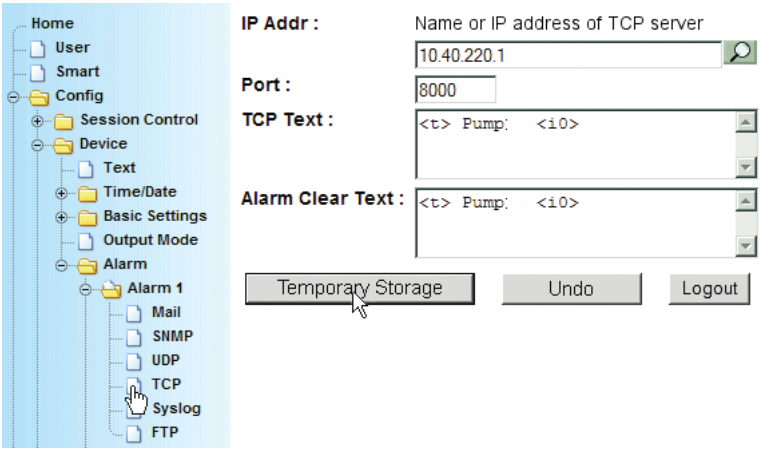
Required access rights: *Administrator*



Send the settings to the Web-IO by clicking on the *Temporary Storage* button.

Under *Config >> Device >> Alarm >> Alarm1 >> TCP* the TCP-specific configurations can then be entered.

Required access rights: *Administrator*



IP Addr
Here you enter the IP address of the network station you want to receive the alarm message.

Port
In this field you enter the local TCP port on which the server application should receive the TCP alarm messages.

TCP Text
Just as with the mail and SNMP or UDP function, you can enter here a text which is sent to the TCP server application. To incorporate the state of inputs and outputs or counter states into the text, you can also use the Alarm tags.

Alarm Variable	Description
<dn>	Device Name look at: Config>>Device>>Text)
<i>	Input state as bit pattern in hex format
<i**x**>	State of Input No. **x** (ON/OFF),
<in**x**>	Name of Input No. **x**
<o>	Output state as bit pattern in hex format
<o**x**>	State of Output No. **x** (ON/OFF),
<on**x**>	Name of Output No. **x**
<c**x**>	Count state of counter no. **x** in decimal format
<t>	Time stamp with date and time formatted DD.MMM.YYYY hh:mm:ss
<$y>	Year formatted "YYYY"
<$m>	Month formatted "MM"
<$d>	Day formatted "DD"
<$h>	Hour formatted "hh"
<$i>	Minutes formatted "mm"
<$s>	Seconds formatted "ss"
	depending on the Web-IO Type **x** can be between 0 and 23

Alarm Clear Text

As in case of TCP text, you can enter here a freely formulated text for the Alarm Clear message.

Send the settings to the Web-IO by clicking on the *Temporary Storage* button.

Switching outputs for a different Web-IO using TCP

Sending of TCP messages can also be used for sending commands to switch outputs on another Web-IO.

For this the address of the other Web-IO must be entered as *IP Addr* and the local port (normally 80) of the other Web-IO as *Port*.

„localhost“ or „127.0.0.1“ or the Web-IO‘s own address may be entered as the address. The Web-IO then sends the command to itself, which enables switching of its own outputs using the alarms.

Under TCP Text the corresponding command for switching an output is entered.

e.g.

```
GET/outputaccess0?PW=<password>&State=ON&NA=ON&
```

for turning on Output 0 or

```
GET/outputaccess0?PW=<password>&State=OFF&NA=ON&
```

for turning it off.

A linefeed should be added to the command by pressing the Enter key.

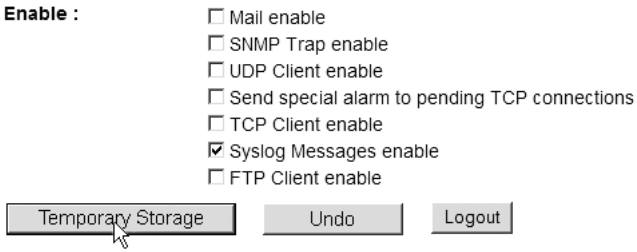
Details on the commands can be found in section 5.6.2

Please note that the trigger for the alarm must be present for min. 1 second for the command to be reliably sent..

5.5.7 Sending alarm messages to a Syslog-Server

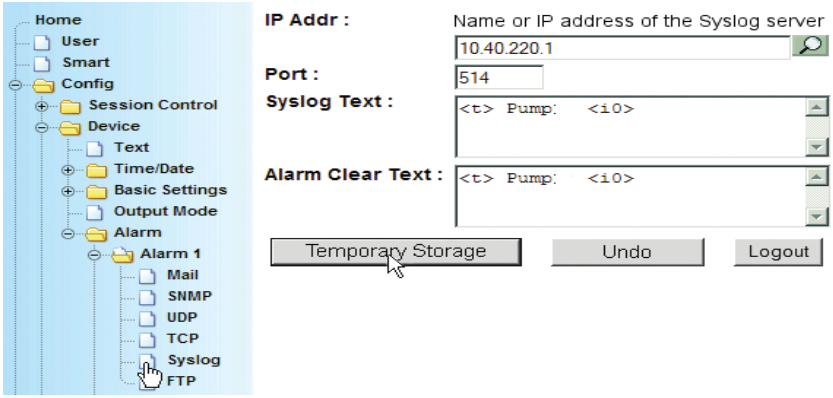
To send a system message to a Syslog server when the alarm condition is met, select *Syslog Message enable* under *Config >> Device >> Alarm >> Alarm1* in the *Enable* area.

Required access rights: *Administrator*



Click on the *Temporary Storage* button to send the settings to the Web-IO.b

Under *Config >> Device >> Alarm >> Alarm1 >>* Syslog you can then enter the Syslog-specific configurations.



IP-Adr
Here you enter the IP address of the Syslog server receiving the Syslog message.

Port
In this field enter the local UDP port on which the Syslog server should receive the Syslog message.

The standard port for Syslog is 514.

Syslog-Text
Just as with the mail and SNMP function, it is possible to enter a text here for sending to the Syslog server. To include the status of inputs and outputs or states of the counters you can also use the alarm variables.

Alarm Variable	Description
<dn>	Device Name look at: Config>>Device>>Text)
<i>	Input state as bit pattern in hex format
<ix>	State of Input No. x (ON/OFF),
<inx>	Name of Input No. x
<o>	Output state as bit pattern in hex format
<ox>	State of Output No. x (ON/OFF),
<onx>	Name of Output No. x
<cx>	Count state of counter no. x in decimal format
<t>	Time stamp with date and time formatted DD.MMM.YYYY hh:mm:ss
<$y>	Year formatted "YYYY"
<$m>	Month formatted "MM"
<$d>	Day formatted "DD"
<$h>	Hour formatted "hh"
<$i>	Minutes formatted "mm"
<$s>	Seconds formatted "ss"
	depending on the Web-IO Type x can be between 0 and 23

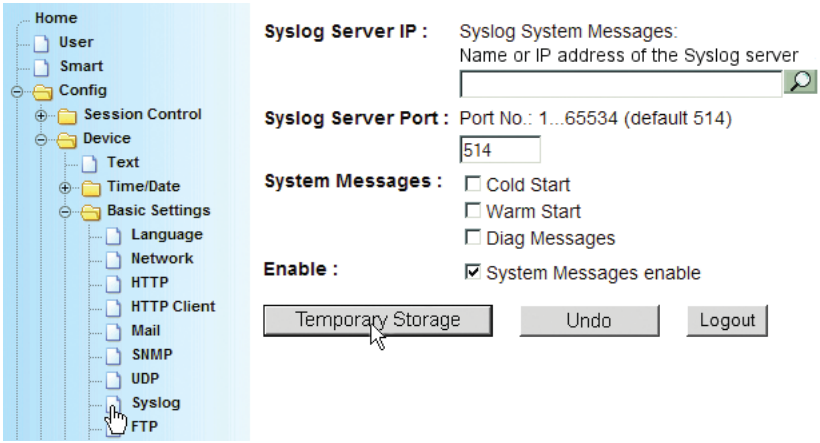
Alarm Clear Text

As in case of syslog text, you can enter here a freely formulated text for the Alarm Clear message.

Send the settings to the Web-IO by clicking on the *Temporary Storage* button.

Finally you must enable the Syslog service.

To do this, select *Config >> Device >> Basic Settings >> Syslog.*

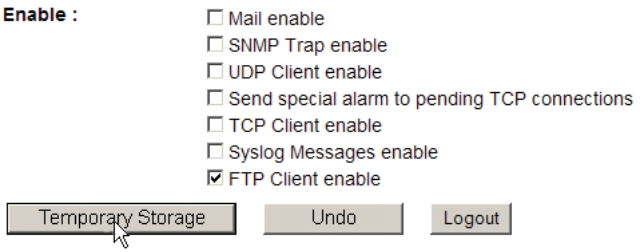


Syslog is enabled by checking *System Messages enable* and clicking on *Temporary Storage*.

5.5.8 Sending alarms via FTP Server

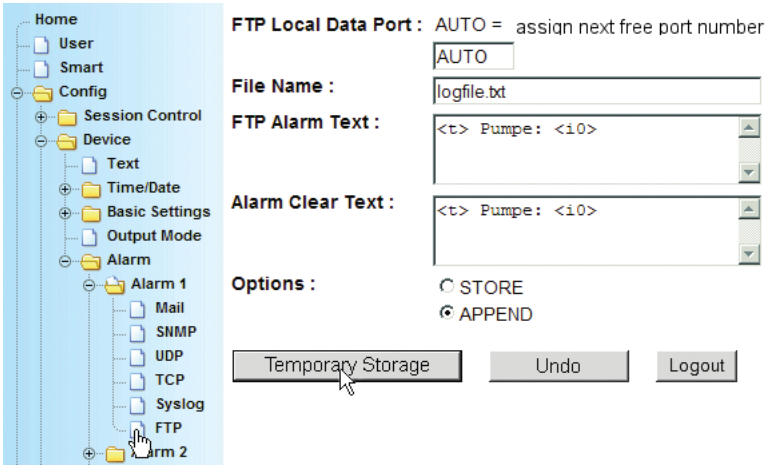
To write a message via FTP to a log file on an FTP server when an alarm condition occurs, highlight *FTP Client enable* in the *Enable* area under *Config >> Device >> Alarm >> Alarm1.*

Required access rights: *Administrator*



Send the settings to the Web-IO by clicking on the *Intermediate store* button.

Under *Config >> Device >> Alarm >> Alarm1 >> FTP* you can then enter the FTP-specific configurations.



FTP Local Data Port
Here you specify the local port for the connection on which the alarm messages are sent by the Web-IO to the FTP server.

If you specify AUTO, the Web-IO uses the next available port.

File Name

In this field you enter the name of the file in which the messages should be saved.

FTP Alarm Text

Just as with the mail and SNMP function, it is possible to enter a text here which is sent to the syslog server. To incorporate the status of inputs and outputs or the counter states of the counters, you can also use the alarm variables.

In addition, text variables can be used in the file name and in the FTP alarm. These are placeholders for status and name of of the individual inputs and outputs, as well as for the counter states of the counters. In addition, placeholders can be used for Date and Device Name.

Alarm Variable	Description
<dn>	Device Name look at: Config>>Device>>Text)
<i>	Input state as bit pattern in hex format
<i**x**>	State of Input No. **x** (ON/OFF),
<in**x**>	Name of Input No. **x**
<o>	Output state as bit pattern in hex format
<o**x**>	State of Output No. **x** (ON/OFF),
<on**x**>	Name of Output No. **x**
<c**x**>	Count state of counter no. **x** in decimal format
<t>	Time stamp with date and time formatted DD.MMM.YYYY hh:mm:ss
<$y>	Year formatted "YYYY"
<$m>	Month formatted "MM"
<$d>	Day formatted "DD"
<$h>	Hour formatted "hh"
<$i>	Minutes formatted "mm"
<$s>	Seconds formatted "ss"
	depending on the Web-IO Type **x** can be between 0 and 23

Alarm Clear Text

As in the case of FTP Alarm Text, here you can enter a freely formulated text for the Alarm Clear message.

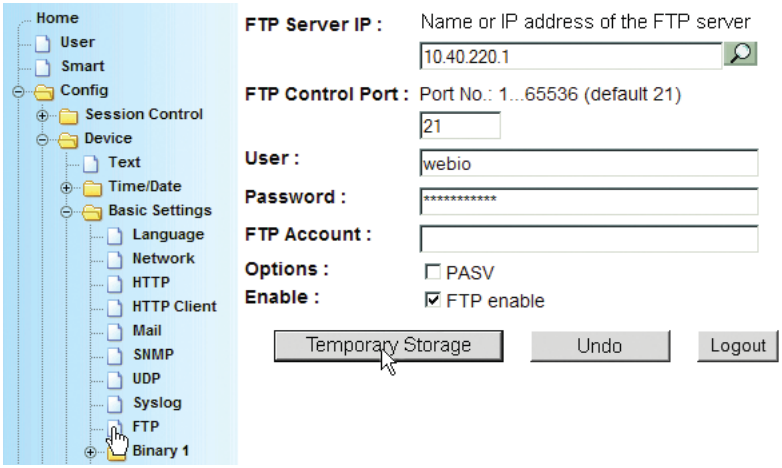
Option

Select *STORE* if you want the file to be completely overwritten each time a message arrives. For a file that expands with each message, select *APPEND*.

Send these settings to the Web-IO by clicking on the *Intermediate store* button

Finally, you must enable the FTP service and configure it.

To do this, use the path *Config >> Device >> Basic Settings >> FTP.*



FTP Server IP
Enter here the IP address of the FTP server on which you want to save the alarm messages.

FTP Control Port
Here the TCP port for the FTP control connection is specified. FTP servers normally use Port 21.

User
Here you enter the name of a registered user with write-access on the FTP server.

Password
User password, stored on the FTP server

FTP-Account
If applicable, enter here the account of the user.

Options
Specify here whether the FTP client should be active or passive. If unsure, consult with your system administrator.

Enable
Highlight *Enable* and click on *Intermediate store* to enable FTP as a service.

Once you have made all your alarm-related entries, activate the new settings with *Config >> Session Control >> LogOut, Save.*

5.5.9 General information about alarms

Multiple and even all send paths at the same time can be used for an alarm. For example, it is no problem to send both an e-mail and an SNMP-Trap.

5.6 Access from your own application

There are two basic ways to access the Web-IO directly from your own applications:

- Access from the browser (HTTP-Request/Java-Applet)
- Access from various high-level languages (TCP/IP-Sockets)

5.6.1 Access from the browser

In its default configuration the Web-IO allows access to the inputs, outputs and counters on three ready-made Web pages.

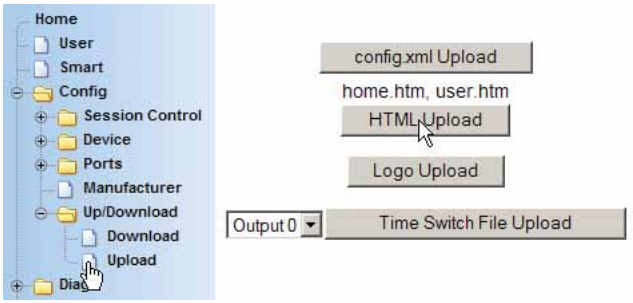
home.htm
- ✓ Dynamic visualization of the IOs and alarms
- ✓ Switching the outputs only after login using the menu tree

user.htm
- ✓ Dynamic visualization of the IOs and the virtual Modbus registers
- ✓ Switching of the outputs after a special login on the User page possible
- ✓ The password can be stored in a cookie so that you do not have to log in each time when opening from the same PC.

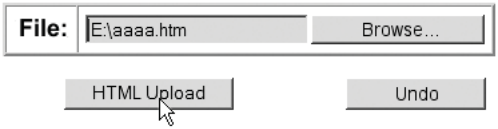
smart.htm
- ✓ Web page optimized for dynamic visualization on smart phones
- ✓ Switching of the outputs possible after a special login
- ✓ The password can be stored in a cookie so that you do not have to log in each time when opening from the same smart phone.

All three pages can be replaced by Web pages you create yourself.

To do this, select *Config >> Up/Download >> Upload.*



and then click on the HTML Upload button.



Select the file you are uploading and confirm by clicking on *HTML Upload*.

The name of the file does not matter. Whether the file is being uploaded as a *home*, *user* or *smart* page will be determined in the first line of the page source text.

```
<user.htm>
<!DOCTYPE html PUBLIC „-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
    <head>
        <meta........
        ..............
```

The Web-IO removes the first line after uploading, before the page is saved (which is why there is no associated closing tag).

Using W&T Tags

As described in section „5.1.4 Labeling and texts“ the Web-IO itself can be used to freely name and label inputs, outputs and counters.

A corresponding display in the browser is accomplished by using W&T tags. W&T tags are placeholders which are replaced

by the Web-IO when sending the Web page to the browser using the stored names. This makes it possible for one and the same Web page to have a different appearance in different Web-IOs.

These tags consist of `<w&t_tags=` and the actual function invocation.

`<w&t_tags=time>`

for example shows the current system time and the date in the browser.

There is a special feature of the W&T tags which show the status of the outputs and the counter states of the input counters.

`<w&t_tags=ox>` and `<w&t_tags=cx>`

When there is an administrator or operator login the displayed contents (ON/OFF or counter state) has a hyperlink. Clicking on this link changes the state of the outputs or sets the counter to 0.

So that the changed state is then displayed, the browser automatically reloads the Web page.

The following W&T tags are available:

Invoke / Tag	Function
device_name	Inserts the name assigned for the Web-IO in the web site
device_text	Inserts the description defined for the Web-IO in the web site.
input**x**	Inserts the name specified for input no. **x**. **x can be a number between 0-11 and indicates which input the invoke refers to. This applies as well to the output and counter invokes described below.**
i**x**	Indicates the state (ON/OFF) of the input corresponding to **x**.
bci**x**	Represents the background of a table cell depending on the state of Input **x**. OFF = no background color, ON = green.
c**x**	Inserts the counter state of the counter for Input **x** in the web site.
output**x**	Inserts the name specified for Output **x**.
o**x**	Shows the state (ON/OFF) of the output corresponding to **x**. When logging in with Operator or Administrator rights, the state indication is given a hyperlink. Clicking on this link changes the state of the corresponding output and refreshes the web site.
bco**x**	Represents the background of a table cell depending on the state of Output **x**. OFF = no background color, ON = green.
access:xxxx:yyyy	Can be added on a link to the pages home.htm, user.htm or index.htm. When there is an Administrator login multiple outputs are switched at the same time. **xxxx** represents in hex format which outputs are permitted to change. **yyyy** represents in hex format to what state the outputs should be switched.
time	Inserts the system time and data of the Web-IO in the Web site.
reload_button	Inserts a button having a reload function in the Web site.
session	When logging in with Operator or Adminstrator rights, the Web IO generates a Session ID. When the page is opened again this Session ID must be specified as a parameter, so that for example when jumping back through a link the access rights are retained.

An example for use of W&T-Tags

The following HTMP source text shows an example of how to use W&T tags:

Since the W&T tags are replaced by the actual content when first loaded into the Web-IO browser, the Web page must first be saved in the Web-IO.

In this case as a User page.

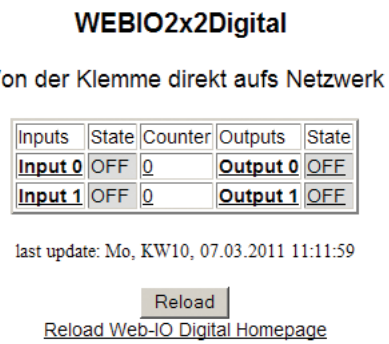
```
<user.htm>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
        <linkrel="stylesheet"href="style.css"type="text/css">
```

Damit die Inhalte der W&T-Tags richtig angezeigt werden muss das Sylesheet *style.css* verlinkt werden

```
        <title>W&amp;T Web-IO Home</title>
    </head>
    <body>
        <div align="center">
            <p class="size5"><b><w&t_tags=device_name></b></p>
            <p class="size4"><w&t_tags=device_text></p>
            <table border="2">
                <tr>
                    <td>Inputs</td>
                    <td>State</td>
                    <td>Counter</td>
                    <td>Outputs</td>
                    <td>State</td>
                </tr>
                <tr>
                    <th><w&t_tags=input0></th>
                    <td <w&t_tags=bci0>><w&t_tags=i0></td>
                    <td><w&t_tags=c0></td>
                    <th><w&t_tags=output0></th>
                    <td <w&t_tags=bco0>><w&t_tags=o0></td>
                </tr>
                <tr>
                    <th><w&t_tags=input1></th>
                    <td <w&t_tags=bci1>><w&t_tags=i1></td>
                    <td><w&t_tags=c1></td>
                    <th><w&t_tags=output1></th>
                    <td <w&t_tags=bco1>><w&t_tags=o1></td>
                </tr>
            </table>
```

```
            <p>last update: <w&t_tags=time></p>
            <form action="home.htm" method="GET" ><w&t_tags=reload_button></form><br>
            <a href="index.htm<w&t_tags=session>" target="_top">Reload Web-IO Digital
            Homepage</a>
        </div>
    </body>
</html>
```

The page looks like this in the browser:


WEBIO2x2Digital
Von der Klemme direkt aufs Netzwerk
Inputs State Counter Outputs State
Input 0 OFF 0 Output 0 OFF
Input 1 OFF 0 Output 1 OFF
last update: Mo, KW10, 07.03.2011 11:11:59
Reload
Reload Web-IO Digital Homepage

Links and backlinks

Some applications may make it necessary to link from the User or Home page of the Web-IO to a Web page which exists on a different HTTP server.

If then the backlink causes navigation back to a Web-IO page, this is done using standard user access rights, even if there was previously an administrator or operator login.

To enable a backlink with the higher access rights we provide the <w&t_tags=session> tag. This tag is a placeholder for the session ID.

The corresponding link would then have to be inserted into the page as follows:

```
<ahref="http://wut.hs2/webioback.htm<w&t_tags=session>"target="_top">Testlink</a>
```

When the link is opened the Web-IO replaces the sessions tag with the session ID and sends this to the newly opened Web page.

To use the session ID for a backlink a small JavaScript must be inserted into the external page which receives the session ID when the page is opened and uses it in the backlink.

```
<html><head>

<script language="JavaScript">
session=window.location.search

function setbacklink() {
 var backlinkurl = "http://172.16.232.17/index.htm";
 var backlink = backlinkurl.concat(session);
 window.location.href = backlink;
}
</script>
```

Im Body-Bereich der Seite kann dann die Funktion *setbacklink* als Hyperlink eingesetzt werden.

```
</head>
    <body>
        Es wird nun mit der übergebenen Session eine Testseite aufgerufen.
        <a href="javascript:setbacklink();"  >Aufruf Testseite</a>
    </body>
</html>
```

This ensures that after jumping back to the Web-IO the access rights level is maintained.

Dynamic Web pages with JavaScript and AJAX

The current status of inputs, outputs and counters can be requested from the Web-IO using HTTP requests.

The following HTTP requests are supported by the Web-IO:

Commands	Parameter	Description
input**x**	?PW=**password**& Password must be filled with the Administrator password. If no password was assigned, enter "PW=&". ***(applies to all commands!!)***	Input status request **x** can be a value between 0-11 and indicates the input. The reply from the Web-IO is a string beginning with "inputx;" followed by the input status: ON = Signal on input and OFF = no signal on input If x is completely omitted, the Web-IO returns a bit pattern corresponding to the input signals in hex format.
counter**x**	?PW=**password**&	Counter value request **x** can be a value between 0-11 and indicates the input. The reply from the Web-IO is a string beginning with "counter**x**;". The counter state of the selected counter is attached in decimal format.
output**x**	?PW=**password**&	Output status request **x** can be a value between 0-11 and indicates the output. The reply from the Web-IO is a string beginning with "outputx;" followed by the input status: ON = Signal on input and OFF = no signal on input If x is completely omitted, the Web-IO returns a bit pattern corresponding to the input signals in hex format.
outputaccess**x**	?PW=**password**& [Mask=XXXX&] State=**ON**/**OFF**/**YYYY**& [NA=ON&] ON: Output = 1, OFF: Output = 0, XXXX: Hex value between 0000 and 0FFF corresponding to the bits to be set YYYY: Hex-Wert between 0000 und 0FFF corresponding to the output bit map.	Setting one or more outputs **x** can be a value between 0-11 and indicates the output to be set. The reply from the Web-IO is a string beginning with "output;" followed by a bit map corresponding to the output signals, in hexadecimal format. Specifying the mask is optional. If mask is not sent, the outputaccess command applies to all outputs. "NA=ON" is an option for causing the Web-IO to send no reply to the outputaccess command.
counterclear**x**	?PW=**password**&	Resets a counter to 0. **x** can be a value between 0-11 and indicates which output to set. The reply from the Web-IO is a string beginning with "counter**x**;". The new counter state of the selected counter is attached in decimal format. If **x** is not specified, all 12 counters are reset to 0. In this case there is no reply.
errorclear	?PW=**password**&	Clears the error report in the Web-IO. See section *Diagnostics and Testing*

Commands	Parameter	Description
time	?PW=**password**&	Returns the system time of the Web-IO in format: DD.MM.YYYY,hh:mm:ss. D=day, M=month, Y=year, h=hour, m=minute, s=second
settime	?PW=**password**& time=**DD.MM.YYYY,hh:mm:ss**&	Sets the system time of the Web-IO to the value transmitted with **time**
diagnosis	?PW=**password**&	Requests the status of the diagnostic memory. The Web-IO returns: diagnosis;0000;00000000;00000000;00000000 The four-digit value represents the number of stored messages. For the 8-digit hexadecimal values each set bit represents one of the 92 possible messages.
diagnosis**x**	?PW=**password**&	**x** indicates the index for one of the currently stored messages. The Web-IO returns the corresponding message text. **x** must not be larger than the number of currently saved messages.
diaglist**x**	?PW=**password**&	Returns the messages for the individual message bits (max. 64)
diagclear	?PW=**password**&	Clears the message memory
errorclear	?PW=**password**&	Clear load errors and re-enables the affected outputs See section *Diagnostics and Testing*

An example for use of HTTP requests

The following example shows how you can construct a self-refreshing Web page for the Web-IO using JavaScript and HTTP requests (AJAX). First the purely HTML section of the Web page, which serves in essence as a display foundation for AJAX:

```
<user.htm>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=windows-1250">
    <title>Web-IO Digital, User</title>
    <style type="text/css">
        * { font-family:arial; }
        table { font-size:14px; }
        .borderLeft { border-left:1px solid #000000; }
        .button { font-size:9px; width:40px; }
        .ce { text-align:center; }
        .description { font-size:18px; text-align:center; }
```

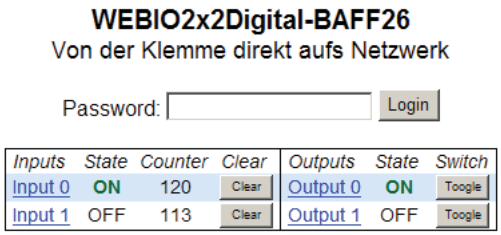
```
        .info { font-size:10px; text-align:center; }
        .italic { font-style:italic; }
        .name { font-size:20px; font-weight:bold; text-align:center }
        .sideSpace { padding-left:5px; padding-right:5px; }
        .table { background-color:#d6e8ff; border-collapse:collapse;
                 border:1px solid #000000; }
        .whiteBack { background-color:#ffffff; }
    </style>
    <script language="JavaScript" type="text/javascript">
```

this is actually the JavaScript section of the page, which is described in greater detail below.

```
    </script>
    </head>
<body onload="CommandLoop();">
<div class="name"><w&t_tags=device_name></div>
<div class="description"><w&t_tags=device_text></div>
<br>
  <form>
    <div align="center">
              <span id="pt"">Password: </span>
             <input id="pw" type="password" name="tfPassword" maxlength="31" size="20"">
              <input  id="lb" type="button" value="Login" onclick="setPassword()">
        </div>
  </form>
<table align="center" class="table">
  <tr class="italic whiteBack">
    <td class="sideSpace">Inputs</td>
    <td class="sideSpace">State</td>
    <td class="ce sideSpace20">Counter</td>
    <td class="ce">Clear</td>
    <td class="borderLeft sideSpace">Outputs</td>
    <td class="sideSpace">State</td>
    <td class="ce">Switch</td>
  </tr>
  <tr>
    <td class="sideSpace"><w&t_tags=input0></td>
    <td class="ce" id="input0">-</td>
    <td class="ce" id="counter0">-</td>
    <td class="ce sidespace">
      <input class="button" onclick="clearCounter(0);" type="button" value="Clear">
    </td>
```

```
    <td class="borderLeft sideSpace"><w&t_tags=output0></td>
    <td class="ce" id="output0">-</td>
    <td class="sideSpace">
      <input class="button" onclick="setOutput(0);" type="button" value="Toogle">
    </td>
  </tr>
  <tr class="whiteBack">
    <td class="sideSpace"><w&t_tags=input1></td>
    <td class="ce" id="input1">-</td>
    <td class="ce" id="counter1">-</td>
    <td class="ce sidespace">
      <input class="button" onclick="clearCounter(1);" type="button" value="Clear">
    </td>
    <td class="borderLeft sideSpace"><w&t_tags=output1></td>
    <td class="ce" id="output1">-</td>
    <td class="sideSpace">
      <input class="button" onclick="setOutput(1);" type="button" value="Toogle">
    </td>
  </tr>
</table>
</body>
</html>
```

Here is how this looks in the browser:


WEBIO2x2Digital-BAFF26
Von der Klemme direkt aufs Netzwerk
Password:
Login
Inputs State Counter Clear Outputs State Switch
Input 0 ON 120 Clear Output 0 ON Toogle
Input 1 OFF 113 Clear Output 1 OFF Toogle

The functions shown below are based on two basic techniques:

- Identification and later changing of an HTML object using a unique ID
- Communication with the server (Web-IO) after loading the Web page using HTTP requests

The JavaScript which is part of the Web page uses both these techniques.

First *maxi* and *maxo* are used to determine how many inputs and outputs are to be supported. A variable *applicationsstep* is defined for the various program steps. 500ms is specified as the request interval, and a variable is entered for the password.

```
var maxi = 2;
var maxo = 2;
var applicationstep = 0;
var interval = 500;
variopassword='';
```

Die Funktion *HextoInt* erlaubt die Umrechnung von hexadezimalen Zahlen in Ganzzahlenwerte (wird benötigt, wenn mit mehr als 2 IOs gearbeitet wird, da die Zustandsübergabe dann hexadezimal erfolgt).

```
function HexToInt(HexStr)
{ var TempVal;
  var HexVal=0;
  for( var i=0; i<HexStr.length;i++)
  { if (HexStr.charCodeAt(i) > 57)
    { TempVal = HexStr.charCodeAt(i) - 55;
    }
    else
    { TempVal = HexStr.charCodeAt(i) - 48;
    }
    HexVal=HexVal+TempVal*Math.pow(16, HexStr.length-i-1);
  }
  return HexVal;
}
```

The function *CommandLoop* combines the HTTP requests and sends them to the *DataRequest* function.

```
function CommandLoop()
{   var commandstring ='';
    applicationstep++;
    switch(applicationstep)
```

```
    {   case 1:
    commandstring = 'input?PW=' + iopassword + '&';
        break;
        case 2:
            commandstring = 'output?PW=' + iopassword + '&';
        break;
        case 3:
            commandstring = 'counter?PW=' + iopassword + '&';
            applicationstep = 0;
        break;
    }
    DataRequest(commandstring);
    maintimer = setTimeout("CommandLoop()", interval);
}
```

The *DataRequest* function is the heart of this JavaScript. It receives the HTTP request and sends it to the server (Web-IO). The *DataRequest* function also receives the reply from the Web-IO and sends it to the *updateDisplay* function.

```
    function DataRequest(SendString)
    {   var xmlHttp;
        if( window.ActiveXObject )       // Internet Explorer
        {   xmlHttp = new ActiveXObject( "Microsoft.XMLHTTP" );
        }
        else if(window.XMLHttpRequest )  // Mozilla, Opera und Safari
        {   xmlHttp = new XMLHttpRequest();
        }
        if (xmlHttp)
        {   xmlHttp.onreadystatechange = function()
            {   if (xmlHttp.readyState == 4)
                {   if (xmlHttp.status == 200)
                    {   if (xmlHttp.responseText.length > 0)
                        {   updateDisplay(xmlHttp.responseText);
                        }
                        xmlHttp=null;
                    }
                }
            }
            xmlHttp.open("GET", SendString, true);
            xmlHttp.setRequestHeader("Connection", "close");
```

```
            xmlHttp.setRequestHeader("If-Modified-Since", "Thu, 1 Jan 1970 00:00:00 GMT");
            xmlHttp.send(null);
        }
    }
```

The *updateDisplay* function evaluates the reply from the Web-IO and correspondingly adjusts the browser display. A check is made as to whether the Web-IO reply refers to inputs, outputs or counters. JavaScript uses *document.getElementById(ID)* to identify the objects for changing and adjusts their properties to the actual IO status.

```
function updateDisplay(ReceiveStr)
{   var HexVal;
    var state;
    var ReceiveData = ReceiveStr.split(';')
    // Display Intput state
    if (ReceiveData[ReceiveData.length - 2].substring(0, 1) == 'i')
    {   HexVal = HexToInt(ReceiveData[ReceiveData.length - 1]);
        for (var i = 0; i < 2; i++)
        {   state = false;
            if ((HexVal & Math.pow(2, i)) == Math.pow(2, i))
            {   state = true;
            }
            document.getElementById('input'+i).firstChild.data = ( !state ) ? 'OFF' : 'ON';
            document.getElementById('input'+i).style.color
                                                    = ( !state ) ? '#000000' : '#006600';
            document.getElementById('input'+i).style.fontWeight
                                                        = ( !state ) ? 'normal' : 'bold';
        }
    }
    // Display Output state
    if (ReceiveData[ReceiveData.length - 2].substring(0, 1) == 'o')
    {   HexVal = HexToInt(ReceiveData[ReceiveData.length - 1]);
        for (var i = 0; i < 2; i++)
        {   state = false;
            if ((HexVal & Math.pow(2, i)) == Math.pow(2, i))
            {   state = true;
            }
            document.getElementById('output'+i).firstChild.data = ( !state ) ? 'OFF' : 'ON';
            document.getElementById('output'+i).style.color
```

```
                                                    = ( !state ) ? '#000000' : '#006600';
            document.getElementById('output'+i).style.fontWeight
                                                        = ( !state ) ? 'normal' : 'bold';
        }
    }
    //Display Counter
    if (ReceiveData.length - maxi - 1 >= 0)
    {   if (ReceiveData[ReceiveData.length - maxi - 1].substring(0, 1) == 'c')
        {   for (var i = 0; i < maxi; i++)
            {   document.getElementById('counter' + i).innerHTML
                                            = ReceiveData[ReceiveData.length - maxi + i]
            }
        }
    }
    //Display cleared Counter
    if (ReceiveData[ReceiveData.length - 2].substring(0, 1) == 'c')
    {   document.getElementById('counter'
    + ReceiveData[ReceiveData.length - 2].substring(7, ReceiveData[ReceiveData.length -
    2].length)).innerHTML = ReceiveData[ReceiveData.length - 1];
    }
}
```

The *setOutput* function sends the corresponding command for toggling the selected output to the *DataRequest* function.

```
function setOutput(iNr)
{   var commandstring = 'outputaccess'+iNr+'?PW='+iopassword+'&State=TOGGLE&';
    DataRequest(commandstring);
}
```

The *clearCounter* function sends the corresponding command for clearing the counter state of the selected counter to the *DataRequest* function.

```
function clearCounter(iNr)
{   DataRequest('counterclear'+iNr+'?PW='+iopassword+'&');
}
```

The *setPassword* takes the entered password and writes it to the *iopassword* variable, which is a component of the command string.

```
function setPassword()
{   iopassword = document.getElementById('pw').value;
    document.getElementById('pw').value = '';
}
```

AJAX, which is based on HTTP requests, can only be used if the Web page is loaded from the Web-IO which will be receiving the requests.

The HTTP requests shown here can just as easily be used for other Web technologies, such as PHP.

Using the Web-IO's own Java applets

Instead of AJAX the IO states of the Web-IO can also be dynamically displayed using Java applets which are loaded from a third-party server.

The applets are incorporated using the <applet> tag, whereby the standard parameters *archive* and *code* are used as follows:

```
archive="dio.jar"
code="dio.class"
```

The other parameters are set individually according to the application.

```
name="dio0"
```

The designator *dio* was chosen here randomly and can be replaced with any other. The only important thing is that when opening the applet function using JavaScript the same name is used.

```
codebase="http://<IP-Adresse>"
```

If the Web page from which the applet is loaded is loaded directly from the Web-IO as *user.htm* or *home.htm* you do not need the parameter *codebase*.

```
width="0"      Angabe in Pixel
height="0"     Angabe in Pixel
```

Since the applet does not contain any visual components, *width* and *height* must always be 0.

`mayscript`

This option allows the Java applet to use JavaScript code in the HTML document

In addition there are several other Web-IO specific parameters which can be set which are always inserted in a separate <param> tag:

device
Whole number value between 0 and 255. The *device* parameter is used for indexing when you need to operate more than one Web-IO on a Web page.

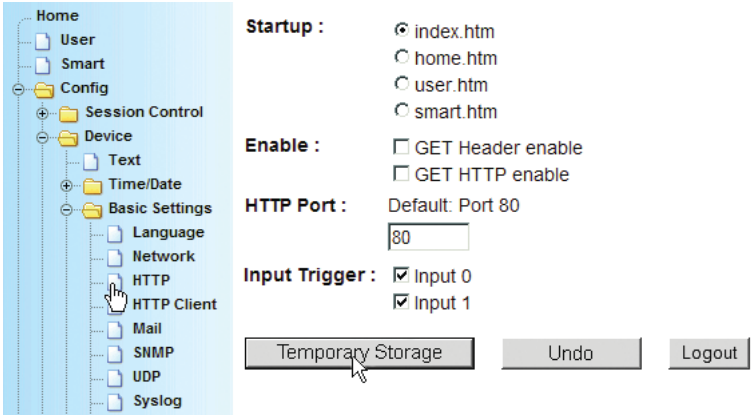
If you are using only one Web-IO, set the value to 0. For all additional Web-IOs incorporated into the Web page the value increments by 1.

showerrors
If errors occur when using the Java applet, this parameter can be used to specify whether an error message should be output in the form of an alert window.

inputpolling
This parameter is used when starting an applet to specify whether the inputs should be checked for their status at an interval set under *pollingrate*.

As the *value* for *inputpolling* you can set *on* or *off*.

To reduce the data load on the network, use the Web-IO navigation tree under *Config >> Device >> Basic Settings >> HTTP* to set the *Input Trigger* for the inputs you want to monitor.



In this case *value* can be set to *off*. Now when the inputs are refreshed there is only a data load if an input has changed.

To activate the changes in the input trigger, the new settings must be saved under *Session Control >> Logout*.

outputpolling
By using this parameter, when the applet is started a determination is made whether the outputs should be checked for their status at the interval set under *pollingrate*.

As the *value* for *outputpolling* you can set *on* of *off*.

counterpolling
By using this parameter, when the applet is started a determination is made whether the counters should be checked for their status at the interval set under *pollingrate*.

As the *value* for *counterputpolling* you can set *on* of *off*.

pollingrate
This parameter specifies, using *value*, at what interval (in ms) inputs, outputs or counters should be refreshed.

Here an example:

```
<applet name="dio0" archive="dio.jar" code="dio.class"
```

```
      codebase="http://172.16.232.11" width="0" height="0" mayscript>
    <param name="device" value="0">
    <param name="showerrors" value="off">
    <param name="inputpolling" value="on">
    <param name="outputpolling" value="on">
    <param name="counterpolling" value="on">
    <param name="pollingrate" value="2000">
</applet>
```

Parameters and values are all case-sensitive.

Using the applet with JavaScript

In order to work with the applet corresponding JavaScript functions must be declared in the header of the Web page.

For read-only access there are 4 functions:

Password protection

```
<script language="JavaScript" type="text/javascript">
<!--
function setPassword()
  { document.applets["dio0"].setPassword('Administratorpasswort');}
```

If an administrator or operator password was set for the Web-IO the *setPassword* function must be opened so that the applet can access the Web-IO. Instead of a static password the user can of course also be prompted by an entry field and the password be sent in that way.

In any case *setPassword* cannot be opened until the Java applet has been started, since otherwise there would be runtime errors.

An example where the password is transmitted using an entry field can be seen when you open the Web page app.htm in the Web-IO.

Read functions

```
function inputChanged( iDevice, iNr, iVal )
  { Programmcode der beim Wechsel an den Inputs ausgeführt wird}
function outputChanged( iDevice, iNr, iVal )
  { Programmcode der beim Wechsel an den Outputs ausgeführt wird}
function counterChanged( iDevice, iNr, iVal )
  { ProgrammcodederbeiÄnderungeinesdenCountersausgeführtwird}
```

The three functions above were opened by the applet when a change in the inputs or outputs or counters is detected. iDevice indicates in which Web-IO a value has changed. *iNr* indicates which input, output or counter has changed. The variable *iVal* represents the current state.

Please note that names of functions are case-sensitive

The following source text shows a small example for dynamic display of Input 0.

```
<html>
  <head>
    <script language="JavaScript" type="text/javascript">
    <!--
    function inputChanged( iDevice, iNr, iVal )
    {   if (iNr == 0 && iDevice == 0)
      { document.getElementById( 'input' + iNr ).firstChild.nodeValue
          = ( !iVal ) ? 'OFF' : 'ON';
      }
      else
      { iVal = 0;
      }
    }
    function setPassword()
    {   document.applets["dio0"].setPassword( 'wut');
    }
    //-->
    </script>
  </head>
  <body>
    <applet name="dio0" archive="dio.jar" code="dio.class"
      codebase="http://172.16.232.11" width="0" height="0">
```

```
        <param name="device" value="0">
        <param name="showerrors" value="off">
        <param name="inputpolling" value="on">
       <param name="outputpolling" value="off">
       <param name="counterpolling" value="off">
       <param name="pollingrate" value="1500">
    </applet>
    <p>State of Input 0:</p>
    <p id="input0">OFF</p>
    <script language="JavaScript" type="text/javascript">
    <!--
      setPassword();
    //-->
    </script>
  </body>
</html>
```

The example contains only the function *inputChanged*, since only *inputpolling* was enabled in the applet tag area.

In any case *setPassword* cannot be opened until the Java applet has been started, since otherwise there would be runtime errors.

The following applies to processing inputs, outputs and counters: If the corresponding polling is enabled, the associated JavaScript function must be declared in the HTML text so that runtime errors are avoided.

If you want the password to be automatically transmitted when the Web page is opened, opening of the setPassword function must be located behind the applet tag.

In this case the password is stored unencrypted (readable) in the source text. For security reasons we therefore recommend password prompting with an entry field.

Write functions

To also permit individual access to the outputs the applet provides corresponding functions.

```
function setOutput( iDevice, iOutput, iValue )
        {   iOut = 0;
            iOut |= Math.pow( 2, iOutput );
            document.applets["dio"+iDevice].outputAccess( iOut, iValue );
        }
```

If you wish to use JavaScript to access the outputs, preceding functions should be inserted into the header of the Web. *iDevice* indicates on which Web-IO an output will be switched. *iOutput* indicates which output to set, and *iValue* is set to *0x0FFF* if the output is supposed to be turned ON. *0x000* corresponds to OFF .

Within the function the JavaScript command *document.applets["dio"*+iDevice*].outputAccess(iOut, iValue)* is used. If you want to switch more than one output at a time, you can use this command in your own functions. The variable *iOut* is a whole number 16-bit value. Using for example mathematical functions or sending constants the bits which are supposed to be changed can be set to 1. In *iValue* you specify for the individual bits whether the corresponding output should be 0 (=OFF) or 1 (=ON).

Example in hex format:

iOut = 0x0013, iValue = 0x0101

In this case Outputs 0 and 4 are set to ON and Output 1 to 0. All other outputs remain unchanged.

Another function is provided for clearing counters.

```
function clearCounter( iDevice, iValue )
        {   iVal = 0;
            iVal |= Math.pow( 2, iValue );
            document.applets["dio"+iDevice].counterClear( iVal );
        }
```

When opening this function *iValue* indicates which of the counters will be cleared.

A detailed example for the use of the Java applet is found on the Web page app.htm, which can be opened in the Web-IO.

Open: http://<ip-adresse>/app.htm

5.6.2 Access from various high level languages

The Web-IO offers two ways to access using TCP/IP sockets.

Access via:

- Command strings **ASCII**
- Binary structures **BINARY**

Command strings ASCII
Depending on the configuration the Web-IO works in this mode as a TCP server, TCP client or UDP peer.

By exchanging simple command strings you can read the inputs and counters and set the outputs.

Outputs are normally read by polling. Event-driven evaluation is only possible if appropriately configured.

Advantages:

- Since all high level languages such as Visual Basic, Visual Basic for Applications, Delphi, C, C++ ...etc. permit uncomplicated working with string variables, even unskilled programmers can quickly obtain the desired results.
- Under TCP Port 80 (otherwise HTTP) is used, which is not blocked by most firewalls. This means that generally nothing in the network infrastructure needs to be modified.
- The combination of using string variables and TCP port 80 makes integration into PHP and CGI scripts quite simple. PLCs and other microcontrollers can be accessed if there is a network connection.

Disadvantage:

- Only conditionally event-controlled programming is possible. This makes it of limited use for time-critical applications.

Summary: Access via command strings speeds up development time for applications where processing speed does not have the greatest priority.

Binary structures BINARY
In this mode the Web-IO can operate as a TCP client, TCP server or UDP peer. Access can be password protected.

The Web-IO defines binary structures for the various functions such as reading inputs, setting outputs etc. Access takes place solely by exchanging these structures.

Advantages:

- Rapid communication through use of structures. The required status values are present at a defined point and no strings have to be checked for their content.
- Event-controlled communication enables the application to respond immediately to changes on the inputs.
- High application security is possible using password protection.

Disadvantage:

- In high level languages which do not support binary structures you must use a workaround in the form of binary strings or byte arrays.

Summary: For professional applications which require high processing speed BINARY mode should be used.

Socket programming with command strings
To access the Web-IO from simple application programs you can use TCP or UDP sockets.

Data exchange between the PC and Web-IO then takes place by sending simple command strings.

The command strings consist of:

The initial sequence: `GET /`
The actual command: `<command>`
The separator: `?`
and one or more parameters: `<parameter1>&<...`

All command strings always end with `&`

Example:

```
GET /input&PW=mypassword&
```

This is how the status of the inputs is polled using a Web-IO protected with *mypassword*.

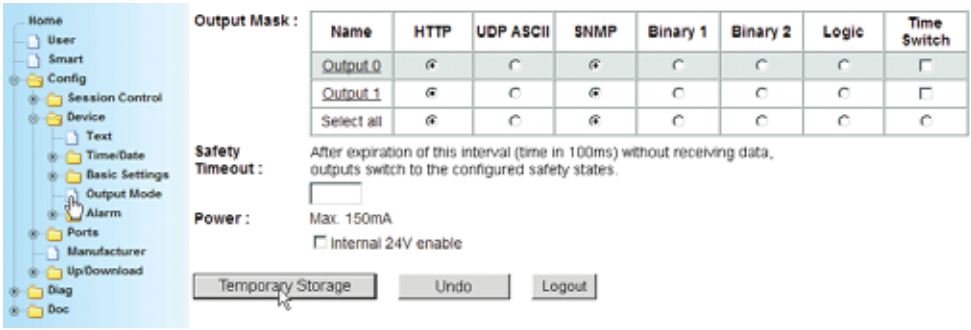
Commands	Parameter	Description
GET /input**x**	?PW=**password**& Password must be filled with the Administrator password. If no password was assigned, enter "PW=&". ***(applies to all commands!!)***	Input status request **x** can be a value between 0-11 and indicates the input. The reply from the Web-IO is a string beginning with "inputx;" followed by the input status: ON = Signal on input and OFF = no signal on input If x is completely omitted, the Web-IO returns a bit pattern corresponding to the input signals in hex format.
GET /counter**x**	?PW=**password**&	Counter value request **x** can be a value between 0-11 and indicates the input. The reply from the Web-IO is a string beginning with "counter**x**;". The counter state of the selected counter is attached in decimal format.
GET /outputaccess**x**	?PW=**password**& [Mask=XXXX&] State=**ON**/**OFF**/**YYYY**& [NA=ON&] ON: Output = 1, OFF: Output = 0, XXXX: Hex value between 0000 and 0FFF corresponding to the bits to be set YYYY: Hex-Wert between 0000 und 0FFF corresponding to the output bit map.	Setting one or more outputs **x** can be a value between 0-11 and indicates the output to be set. The reply from the Web-IO is a string beginning with "output;" followed by a bit map corresponding to the output signals, in hexadecimal format. Specifying the mask is optional. If mask is not sent, the outputaccess command applies to all outputs. "NA=ON" is an option for causing the Web-IO to send no reply to the outputaccess command.
GET /outputaccess**x**	?PW=**password**& State=**ON**/**OFF**/**XXXX**& ON: Output = 1, OFF: Output = 0, XXXX: Hex value between 0000 and 0FFF corresponding to the output bit pattern.	Sets one or more outputs **x** can be a value between 0-11 and indicates which output to set. The reply from the Web-IO is a string beginning with "output;" followed by a bit pattern corresponding to the output signals in hex format.
GET /counterclear**x**	?PW=**password**&	Resets a counter to 0. **x** can be a value between 0-11 and indicates which output to set. The reply from the Web-IO is a string beginning with "counter**x**;". The new counter state of the selected counter is attached in decimal format. If **x** is not specified, all 12 counters are reset to 0. In this case there is no reply.
GET /errorclear	?PW=**password**&	Clears the error report in the Web-IO. See section *Diagnostics and Testing*

Commands	Parameter	Description
GET /time	?PW=**password**&	Returns the system time of the Web-IO in format: DD.MM.YYYY,hh:mm:ss. D=day, M=month, Y=year, h=hour, m=minute, s=second
GET /settime	?PW=**password**& time=**DD.MM.YYYY,hh:mm:ss**&	Sets the system time of the Web-IO to the value transmitted with **time**
GET /diagnosis	?PW=**password**&	Requests the status of the diagnostic memory. The Web-IO returns: diagnosis;0000;00000000;00000000 The four-digit value represents the number of stored messages. For the 8-digit hexadecimal values each set bit represents one of the 64 possible messages.
GET /diagnosis**x**	?PW=**password**&	**x** indicates the index for one of the currently stored messages. The Web-IO returns the corresponding message text. **x** must not be larger than the number of currently saved messages.
GET /diaglist**x**	?PW=**password**&	Returns the messages for the individual message bits (max. 64)
GET /diagclear	?PW=**password**&	Clears the message memory
GET /errorclear	?PW=**password**&	Clear load errors and re-enables the affected outputs See section *Diagnostics and Testing*

TCP communication - the Web-IO as server

First you must determine which of the 12 outputs will be made available for access via command strings.

Select in the navigation tree *Config >> Device >> Output Mode*



Required access rights: *Administrator*

All the outputs that will be used must be activated for *HTTP* (factory default).

If you have changed the output modes, click on the *Temporary Storage* button and then on the *Save and activate all temporary settings* link. Clicking on the *Save* button saves all your settings.

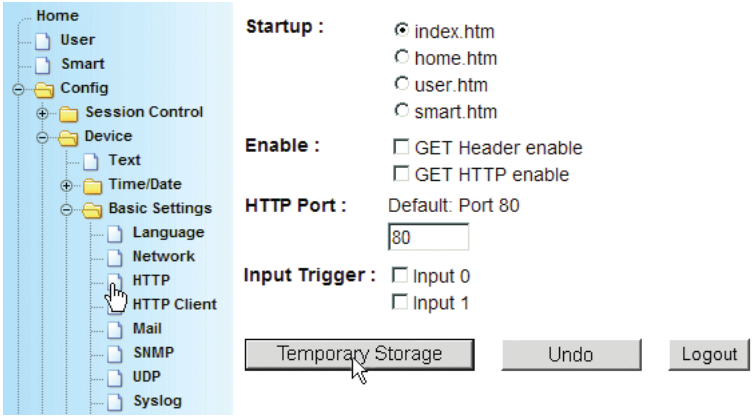
The Web-IO now functions as a TCP server, which results in a three-part program sequence regardless of which programming language is used:

1. Opening of the TCP connection
The application always acts as a client and therfore opens the TCP connection.

The socket interface of the Web-IO is broadly derived from HTTP protocol. This means that the **Listen port** is factory set to **TCP-Port 80** and corresponds to the configured HTTP port.

To use a different port, it is possible to change the HTTP port using Config >> Device >> Basic Settings >> HTTP.

m Required access rights: *Administrator*



Under HTTP port you may enter a TCP port for HTTP access which is different from the standard Port 80.

If the HTTP port is changed, the Web-IO will only be reachable from the browser by specifying the selected HTTP port.

Example for HTTP Port = 85.

This applies to all configuration accesses as well.

As a protection against blocked TCP connections, the Web-IO has a timer set for Port 80 which automatically closes the connection 30 seconds after it is opened if no syntactically correct commands have been received. After a connection has been successfully opened, the application should therefore always send for example an output query to the Web-IO.

2. Communication between Web-IO and application

Normal communication takes place in polling mode. This means: the client application uses the command strings to request the desired values and set the outputs.

Example 1: Query the status of Input 3, where there is a +12 V signal present. No administrator password was assigned for the Web-IO.

The application sends the following string to the Web-IO:

```
GET/input3?PW=&
```

The Web-IO responds as follows:

```
input3;ON
```

Example 2: Set outputs 3,4,7,9. In binary this is 0000 0010 1001 1000, which corresponds to 0298 hex.

The administrator password for the Web-IO is: „blue“

The application sends the following string to the Web-IO:

```
GET/outputaccess?PW=blue&State=0298&
```

The Web-IO responds as follows:

```
output;0298
```

For the Web-IO 24xDigital 8-character hexadecimal values!

The Web-IO works with null-terminated strings! This means that everything the Web-IO sends to the application ends with a chr(0) *character*

For Web-IOs with relay outputs you must maintain a pause of at least 200ms between two switching operations on one output!

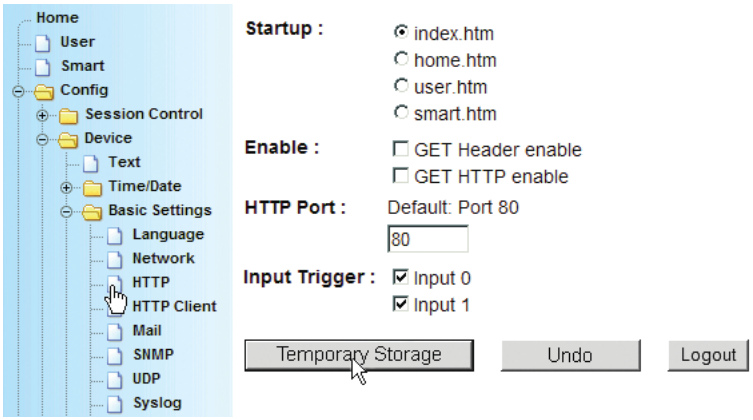
Event-driven messaging

Some applications make it necessary to report changes on certain inputs automatically over the existing TCP connection.

One reason for this is to reduce the data load on the network. If the inputs are polled at a cyclical interval, this will create a continuous data load. There is also the risk that changes on the inputs will not be detected if they occur between intervals.

To set the conditions for event-driven messaging, go to the navigation tree and select *Config >> Device >> Basic Settings >> HTTP*

Required access rights: *Administrator*



Input Trigger

Here you specify which inputs are to be monitored for a state change. If there is an existing connection, the Web-IO automatically sends the new input state to the application.

Include a header

The Web-IO can if needed precede the sent string with a header which contains the IP address and the name.

Example:

- The IP address of the Web-IO is 172.16.232.17.
- „W&T Web-IO 12xDigital“ was entered as the device name under *Config >> Device >> Text* .
- The application polled the inuts using *GET /input?PW=&*.

The Web-IO returns:

```
172.16.232.6;W&TWeb-IO12xDigital;input;0000
```

If GET Header enable was activated, the header precedes all data transmission from the Web-IO.

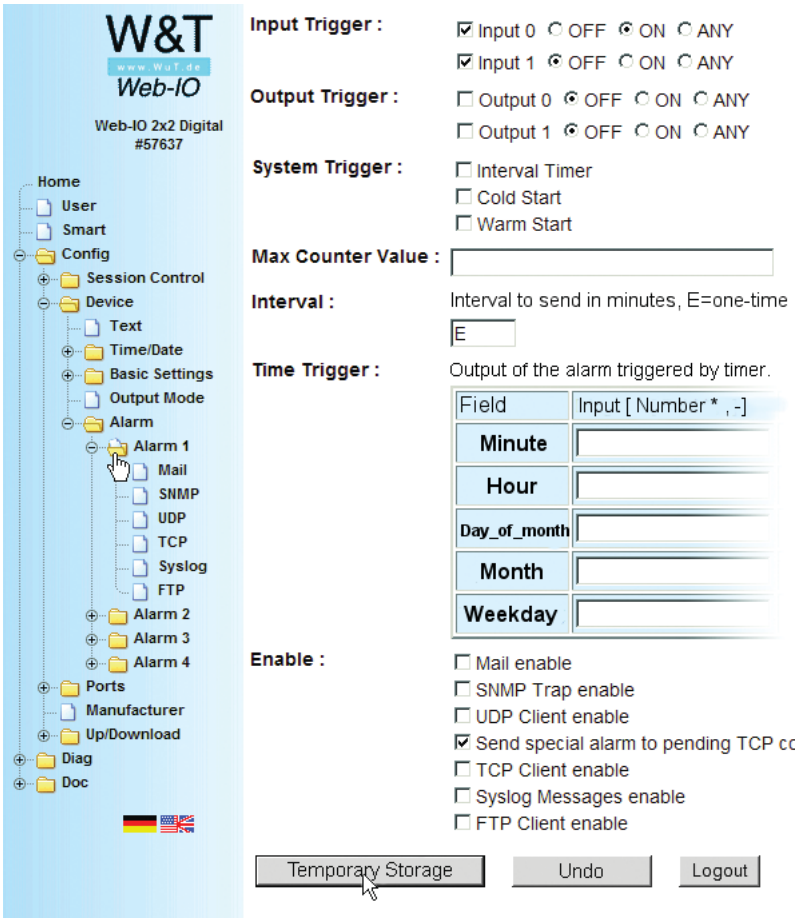
If all conditions have been specified, activate the new setting using *Config >> Session Control >> LogOut.*

Messaging with defined input patterns
In some applications it is practical to get the state of the inputs only when certain input patterns occur.

The messaging method described here can be used in tandem with normal event-triggered operation; in practice however this does not offer any advantages, since in this case there will be a message whenever there is a change. This means you would receive double messaging whenever the specified input pattern occurs.

To set the conditions for a TCP alarm message when certain input patterns occur, go to the navigation tree and select *Config >> Device >> Alarm >> Alarm 1*

Required access rights: *Administrator*



Input Trigger

Here you specify which inputs are to be monitored for which state. If you are activating multiple inputs in an alarm for monitoring, there is an AND-operation on the conditions.

Example:

In Alarm 1 Input 1 was activated for ON, Input 2 for OFF.

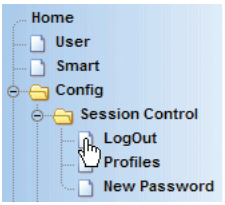
An alarm is only sent to the TCP application if both conditions are met. In this case: Input 1 = ON, Input 2 = OFF.

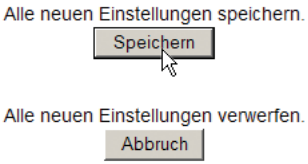
Interval
To send alarms over an existing TCP connection it is practical to set the *E* parameter here so that only one alarm message is sent when the alarm condition occurs.

Enable
Activate the *Send input to pending TCP connections* in the *Enable* area.

Pass the selected alarm conditions to the Web-IO by clicking on the *Temporary storage* button.

When all the alarm conditions have been specified, activate the new settings using *Config >> Session Control >> LogOut*





Clicking on the *Save* button restarts the Web-IO with the new parameters.

3. Close the TCP connection
Here the Web-IO deviates from HTTP protocol. In contrast to HTTP protocol, it is always the client application which closes a connection.

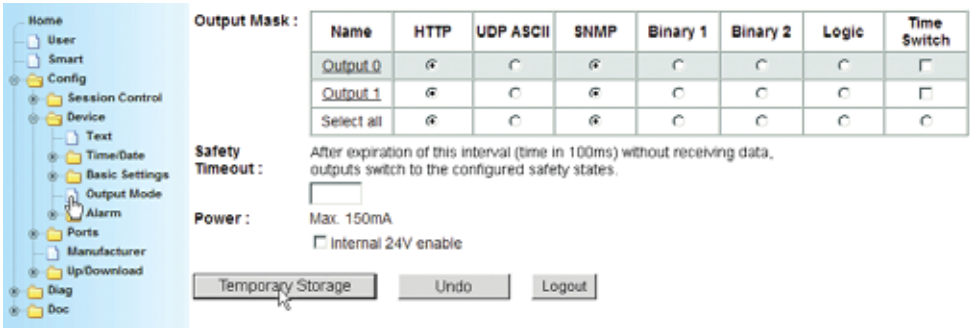
Exception: When it receives a defective command string, the Web-IO immediately closes the connection

TCP communication - the Web-IO as Client
In some applications it is necessary to use the Web-IO as client. In this case the Web-IO opens a TCP connection to a server application. The trigger for opening the connection may be a

status change on one of the inputs, or a time-based opening of a connection may be configured.

First you must specify which of the outputs should be available for access via command strings after the connection is opened.

In the navigation tree select *Config >> Device >> Output Mode*



Required access rights: *Administrator*

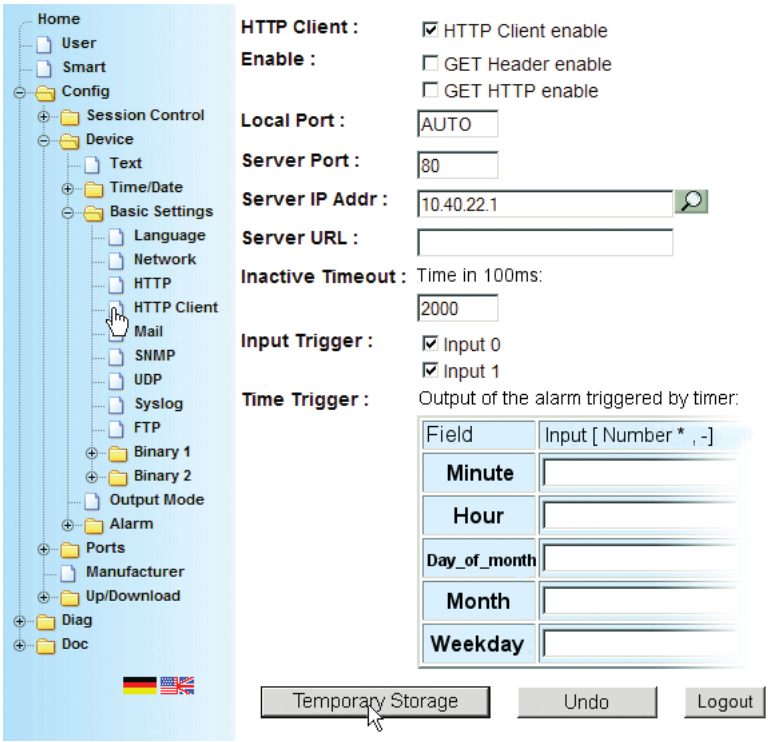
At least one or all outputs used must be activated for *HTTP*.

If you have changed the output modes, click on the *Temporary storage* button.

Then you must configure and activate the Web-IO for TCP-client communication.

In the navigation tree select *Config >> Basic Settings >> HTTP-Client*

Required access rights: *Administrator*



HTTP Client
This field must be acvitated for the Web-IO to operate as a TCP-client.

Enable
Activate this field if you want to precede the data sent by the Web-IO to the server application with a header.

Example:

- The IP address of the Web-IO is 172.16.232.17.
- As a *Device Name* „W&T Web-IO 12xDigital“ was entered under *Config >> Device >> Text*.
- The server application queried the inputs using *GET / input?PW=&*.

The Web-IO sends:

172.16.232.17;W&TWeb-IO12xDigital;input;0000

If *GET Header enable* was activated, the header precedes all the data sends from the Web-IO. In server applications that can accept more than one connection, this makes it very simple to distinguish from where the data is coming.

Local Port
Enter the desired local port in the *Local Port* field. The factory default setting is *AUTO* and the Web-IO uses the next available port.

Server Port
Here you enter the port on which the server application accepts the connection.

Server IP-Adress
Here you enter the IP address of the PC to connect to.

Inactivity Timeout
After the timeout time entered here the Web-IO automatically closes the connection to the server. The timeout time starts as soon as there is no more data traffic on the connection.

Input Trigger
Here you highlight all inputs that are supposed to trigger a connection opening or input status packet for the server application when there is a status change.

Time Trigger
In this area you can configure a time-controlled connection opening.

Field	Input [Number * , -]	Range of values
Minute	0	0-59
Hour	0-23/2	0-23 (0 is midnight)
Day_of_Month	*	1-31
Month	*	1-12
Weekday	1-5	0-6 (0 is Sunday)

The syntax for the time setting is derived from the Cron format used in Linux/Unix. Multiple entries separated by commas can be entered in one field.

The „*“ stands for any permissible whole-number value. Example: * in the Month field stands for every month.

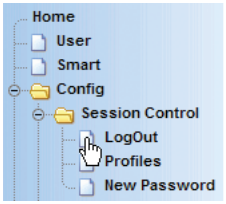
„-“ can be used to use time spans as triggers. Example: 1-5 in the Weekday field means Mo - Fr

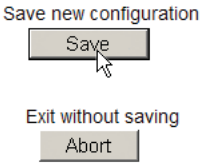
If a „/“ is placed after the actual value or value range, a cycle value can be indicated behind it. Example: 0-23/2 in the Hour field means an alarm every two hours, even if the conditions in the other fields are met. Wildcards („*“) are not allowed together with „/“. (*/2 would not bring a reliable result)

The fields filled in as above therefore mean: every month, on every Monday through Friday, every two hours on the hour.

Send the selected settings to the Web-IO by clicking on the *Intermediate store* button.

Once all your settings have been made, activate them using *Config >> Session Control >> LogOut*





After clicking on the *Save* button the Web-IO will now run as a TCP client.

Communication between Web-IO and application

1. Opening the TCP connection
The TCP connection is opened by the Web-IO according to the configured trigger conditions.

After the connection has been successfully opened, the Web-IO sends the input status in format

```
input;0000
```

to the server application.

The 4-place numerical value (8-place for the 24x Web-IO) represents the input status in hex format.

The Web-IO uses null terminated strings!
This means that everything the Web-IO sends to the application ends with a 0 character.

For Web-IOs with relay outputs you must maintain a pause of at least 200ms between two switching operations on one output!

2. Communication between Web-IO and application
Whenever there is an existing connection, the server application can communicate with the Web-IO using command strings.

Example 1: Query the status of Input 3, on which a +12 signal is present. No administrator password has been assigned to the Web-IO.

The application sends the following string to the Web-IO:

```
GET/input3?PW=&
```

The Web-IO returns with:

```
input3;ON
```

Example 2: Set outputs 3,4,7,9. In binary this is 0000 0010 1001 1000, which corresponds to 0298h.

The administrator password for the Web-IO is: „blue“

The application sends the following string to the Web-IO:

```
GET/outputaccess?PW=blau&State=0298&
```

The Web-IO returns with:

```
output;0298
```

The Web-IO uses null terminated strings!
This means that everything the Web-IO sends to the application ends with a 0 character.

3. Closing the TCP connection
The Web-IO automatically closes the connection when the set timtout time after the last data communication between Web-IO and server application has elapsed.

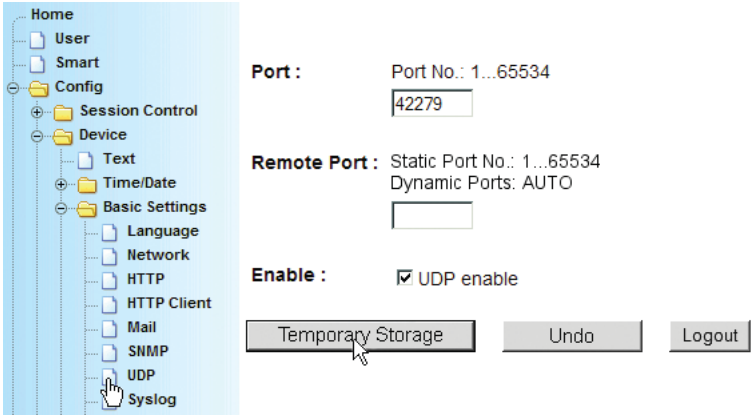
Naturally the server application may close the connection at any desired time.

UDP communication
First you need to activate the Web-IO for communication via UDP command strings and enter the local port.

Select in the navigation tree *Config >> Basic Settings >> UDP*

Required access rights: *Administrator*

In the Port field enter the desired local port. The factory setting is Port 42279. Check UDP enable.



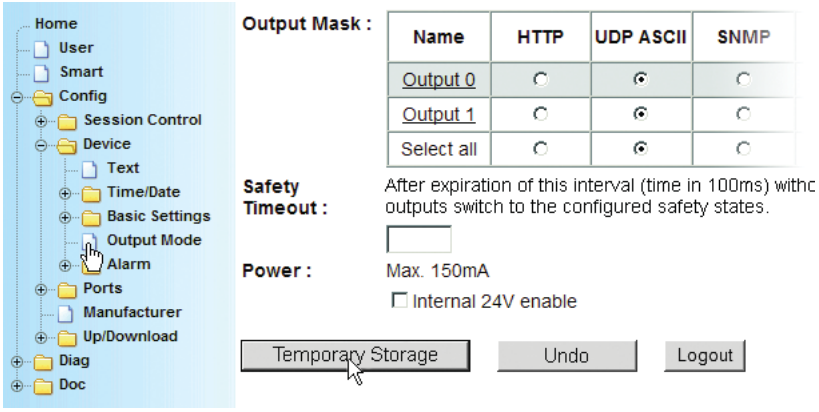
Click on the *Temporary Storage* button to pass the selected settings to the Web-IO.

You must also specify which of the 12 outputs you want to make available for access via UDP using command strings.

Select in the navigation tree *Config >> Device >> Output Mode.*

Required access rights: *Administrator*

All the outputs used must be activated for *UDP ASCII.*



If you have changed the output modes, click on the *Temporary Storage* button and then on the *Save and activate all*

temporary settings link. Clicking on the *Save* button saves all your settings.

Communication between Web-IO and the application
In contrast to TCP, there is no communication between the Web-IO and the other communications partner when communicating via UDP protocol. Data are exchanged using datagrams.

Normal communication takes place in polling mode. This means the application uses command strings to request the desired values and set the outputs.

The Web-IO always responds to a poll by returning the IP address and port of the requesting network station.

Example 1: Query the status of Input 3, where there is a signal of +12 V present. No administrator password was asigned for the Web-IO

The application sends the following string to the Web-IO:

```
GET/input3?PW=&
```

The Web-IO responds as follows:

```
input3;ON
```

Example 2: Set outputs 3,4,7,9. In binary this is 0000 0010 1001 1000 , which corresponds to 0298 hex.

The administrator password for the Web-IO is: „blue“

The application sends the following string to the Web-IO:

```
GET/outputaccess?PW=blue&State=0298&
```

The Web-IO responds as follows:

```
output;0298
```

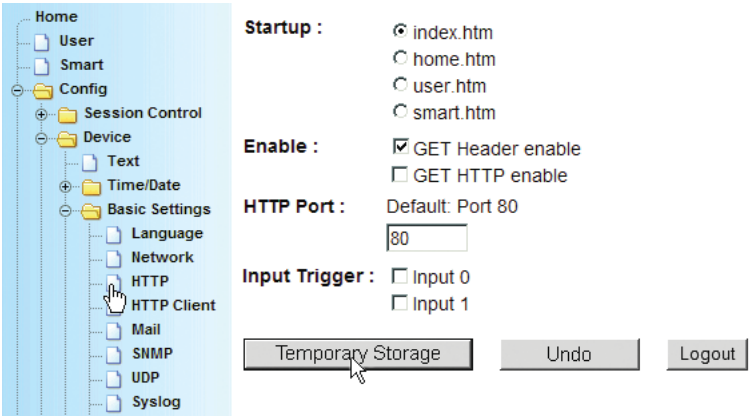
The Web-IO works with null-terminated strings! This means that everything the Web-IO sends to the application ends with a chr(0) *character*

So that under UDP a message can be automatically received when states change, the Web-IO provides a variety of various alarm conditions.

A detailed description of how to use these alarm possibilities can be found in the Alarms *section*

Include a header

The Web-IO can if needed precede the sent string with a header containing the IP address and the name.



To enable a header, go to *Config >> Basic Settings >> HTTP* and select *GET Header enable*. Clicking on *Logout* and *Save* activates sending of the header.

Example:

- The IP address of the Web-IO is 172.16.232.17.
- „W&T Web-IO 12xDigital“ was entered as the device name under *Config >> Device >> Text* „“.
- *GET /input?PW=&* was used by the application to poll the inputs.

The Web-IO returns:

172.16.232.6;W&T Web-IO 12xDigital;input;0000

If GET Header enable was activated, the header precedes all data sent by the Web-IO.

Program-controlled error management
If the Web-IO detects one or more errors while processing its tasks, these are saved and are viewable from the navigation tree under *Diag*.

A more detailed description of this can be found in Section 6.6 Diagnostics and Testing.

In some applications it may be desired that error management is handled not manually by viewing a Web page, but rather errors are automatically evaluated in a program.

For such cases the Web-IO provides several command strings for using TCP or UDP.

Creating a list pf possible error messages

The Web-IO can manage a maximum of 64 different error types. The actual number may vary within this limit depending on the firmware version.

To get an overview of the possible errors, the command

`GET/diaglist`**`x`**`?PW=`**`password`**`&`

can be sent to the Web-IO.

x is replaced by a value between 0 and 64. INstead of *password* enter the Administrator or Operator password.

The Web-IO replies with the corresponding error message.

`diaglist`**`x`**`;`**`errortext`**

Example:

If the application sends (Passwort = „wut“) to the Web-IO

```
GET/diaglist2?PW=wut&
```

the Web-IO responds with

```
diaglist2;FormatfehlerinderDNSAnfrage
```

With a *for next* loop the *diaglist* command can be used to read out all possible error messages and save them as a static list. In binary mode, for example, you can only determine how many errors there are currently. The associated error numbers are also sent. By using the static error list a readable error can be output.

Direct evaluation of the current errors
To get the current error status of the Web-IO you can use the command

```
GET/diagnosis?PW=password&
```

The Web-IO responds with

```
diagnosis;iiii;zzzzzzzz;yyyyyyyy;xxxxxxxx
```

In *zzzz* the Web-IO returns the number of current errors in 4-place hex format.

zzzzzzzz;yyyyyyyy; xxxxxxxx may be ignored for normal, standard applications.

For expert programmers: yyyyyyyy; xxxxxxxx *are two 32-bit values, each in 8-place hex format. THe resulting 64 bits substitute for the 64 possible error types. By using the static error list (see GET /diaglist...) the individual bits can be decoded. The LSB appears to the right in the x range and the MSB to the left in the y range.*

To get the current errors as an error text, the Web-IO provides the command

```
GET/diagnosisx?PW=&
```

Here x is the index for the error in decimal format beginning with 1.

The reply from the Web-IO then looks as follows:

```
diagnosisx;errortext
```

In contrast to the error texts which are read from the Web-IO using *diaglist*, the error text for *diagnosis* may also contain dynamic elements such as IP addresses or port numbers.

Example:

The application sends:

```
GET/diagnosis?PW=wut&
```

The Web-IO responds:

```
diagnosis;0005;00400040;008A0000;00000000
```

The application sends:

```
GET/diagnosis1?PW=wut&
```

The Web-IO responds:

```
diagnosis1;MailServerantwortetnicht.
```

The application sends:

```
GET/diagnosis2?PW=wut&
```

The Web-IO responds:

```
diagnosis2;DasVersendenderMailwurdeabgebrochenundwirdwiederholt.
```

The application sends:

```
GET/diagnosis3?PW=wut&
```

The Web-IO responds:

```
diagnosis3;ZielIP-Adresseunbekannt:172.16.232.8.
```

The application sends:

```
GET/diagnosis4?PW=wut&
```

The Web-IO responds:

```
diagnosis4;WatchdogTimerabgelaufen!
```

The application sends:

```
GET/diagnosis5?PW=wut&
```

The Web-IO responds:

```
diagnosis5;TCPClientAlarm:Servernichterreicht.
```

Clearing the error memory

The list of errors occuring during runtime remains stored in the Web-IO, even if the error is no longer present at the time of query.

To clear the error memory, use the command

```
GET/diagclear?PW=password&
```

The Web-IO responds with

```
diagnosis;0000;00000000;00000000;00000000
```

if there are no current errors.

Special handling of overload errors (TCP only)
The Web-IO can continuously monitor for the load on the outputs. If the maximum output current is exceeded, the corresponding output is turned off as long as the overload is present.

The menu item *Config >> Device >> Outputmode* can also be used to activate continuous turnoff of the outputs. To do this, activate *Load Control enable.*

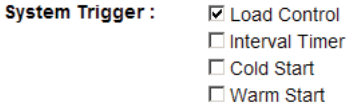
In case of overload the affected output is completely turned off an must be enabled again by the user.

See also Section 6.6 Diagnostics and Testing.

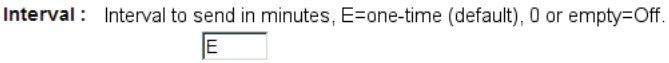
To automatically process overload errors from the application program, it is possible to define a corresponding alarm which sends an error message to the program when a TCP connection is open.

Select *Config >> Device >> Alarm >> Alarm1*

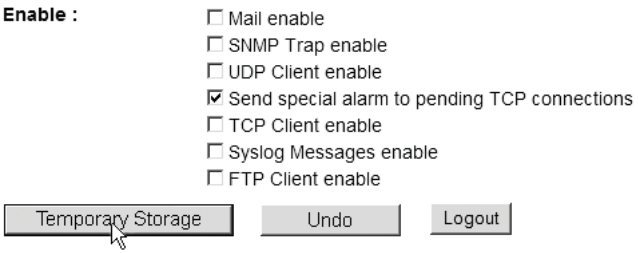
Activate *Load Control* under *System Trigger.*



Enter E as an interval (= one-time).



Under Enable check



and click on Intermediate Storage.

Once all entries pertaining to the alarm have been made, activate the new settings under *Config >> Session Control >> LogOut, Save.*

Now if an overload condition arises, the Web-IO sends the following to the application over the open TCP connection:

```
error;xxxx
```

where *xxxx* is the hex value for the output which was turned off.

Example:

```
error;0001
```

In this case Output 0 is affected.

To turn on the affected outputs again, the application program sends

```
GET/errorclear?PW=password&
```

to the Web-IO.

Now if there is no longer an overload condition, the Web-IO resumes as normal on all outputs.

Please note that the GET HEADER ENABLE *setting under* Config >> Device Basic Settings >> HTTP *also precedes*

the Error and Diag messages with a header. The 0 character is still appended for error messages.

BINARY - socket programs with binary structures
The Web-IO 12 X Digital provides two independent socket accesses, *Binary 1* and *Binary 2*, for carrying out binary data exchange. Both can be used and configured independently of each other.

Whether the Web-IO functions on the respective BINARY socket as a TCP server, TCP client or UDP peer depends on the desired application.

First an overview of which operation mode the Web-IO uses for which application.

- Your own socket application (binary with password protection)
  - TCP-Server
  - TCP-Client
  - UDP-Peer

- Your own socket application, which was originally programmed for the W&T Digital I/O Com-Server 50xxx.
  - TCP-Server
  - TCP-Client
  - UDP-Peer

- Box-to-Box Master
  - TCP-Client

- Box-to-Box Slave
  - TCP-Server

- OPC-Device together with the W&T OPC-Server
  - TCP-Server

Binary socket access and 50xxx-Mode
In this section you will be shown how the Web-IO 12XDigital can be accessed from your own professional applications using sockets with binary structures.

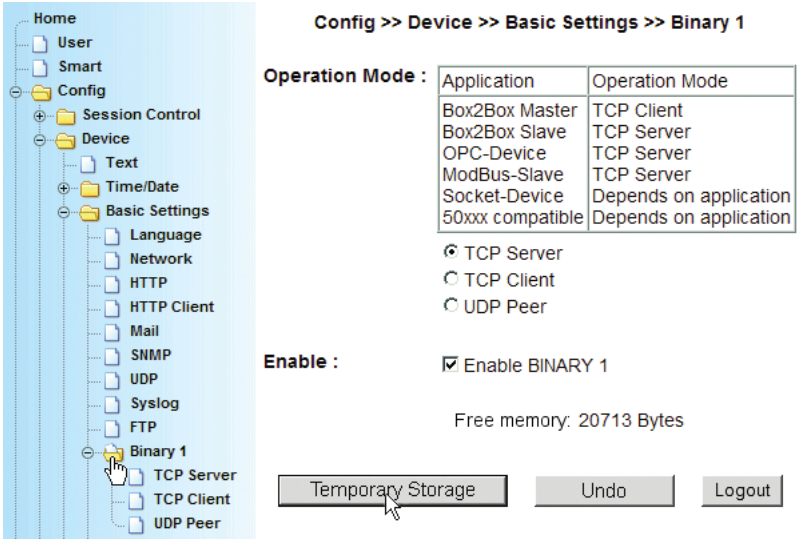
Box-to-box and OPC device modes will be treated in detail in the following sections

Specifying the operation mode

First you must specify whether the Web-IO is to be a TCP client, TCP server or UDP peer in your application.

Select in the navigation tree *Config >> Device >> Basic Settings >> Binary 1* if you want to configure the operation mode for access through *Binary 1*.

Required access rights: *Administrator*



After selecting the desired mode, pass the setting to the Web-IO by clicking on *Temporary Storage*.

Two levels of socket programming are available to the programmer for access from your own application programs.

1. Password protected access
2. 50xxx-Mode compatible with the older W&T Digital I/O Com-Servers

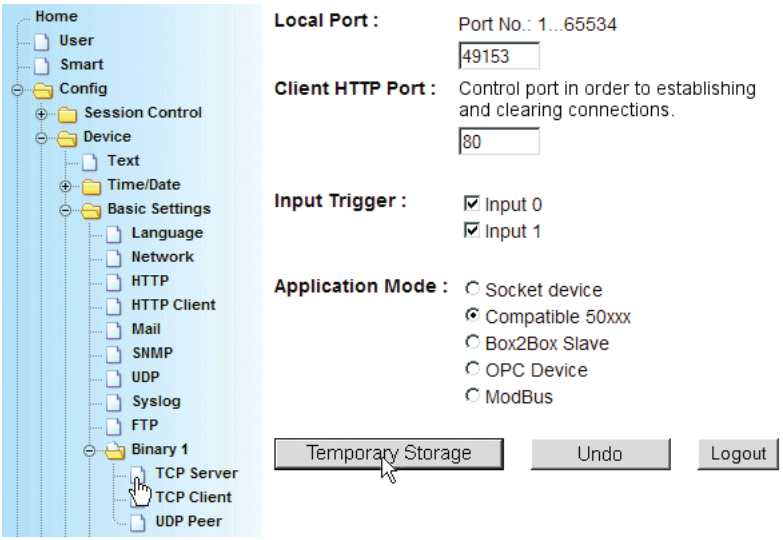
Both levels work with the same binary structures and differ only in the lack of password protection in 50xxx mode.

The Web-IO as socket server

To operate the Web-IO as a socket server, you must make a few additional settings.

Select in the navigation tree *Config >> Device >> Basic Settings >> Binary 1>> TCP Server*

Required access rights: *Administrator*



Local Port

The local port of the Web-IO is factory set for 49153. If your application requires a different local port for the Web-IO, enter the desired port number in the *Local Port* field.

Client HTTP Port

Has meaning only in OPC and Box2Box modes and specifies the HTTP port on which the control connection to the OPC server or slave box should be opened.

Unless otherwised specified, always use Port 80 here.

Input Trigger
Here you check the inputs whose state change will trigger data to be sent to the client application (important for event-triggered applications).

Application Mode
Select here:

- *Default* - If you want access to the Web-IO to be password protected.
- *Compatible 50xxx* - If you want to access applications that were programmed for the older Digital I/O Com-Servers. You can also use this mode for new applications that do not require password protection.

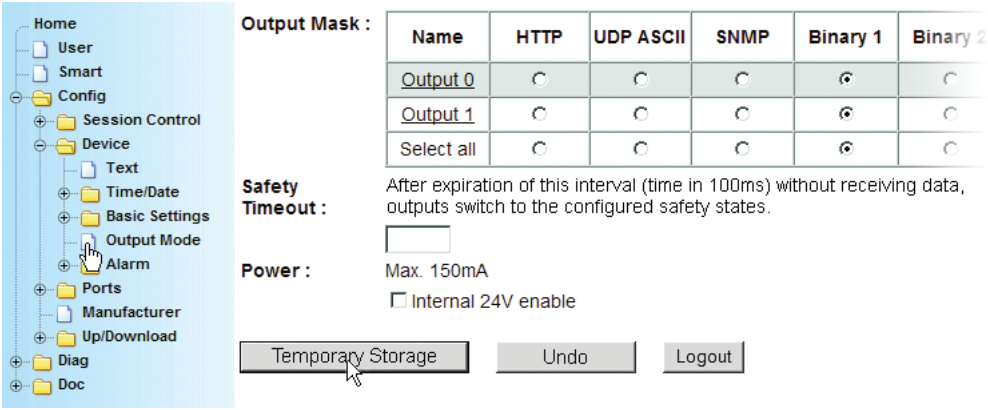
Additional information concerning *Box2Box Slave* and *OPC Device* modes can be found in the corresponding sections.

Once you have made all your settings, send them to the Web-IO by clicking on the *Temporary Storage* button.

In addition you must enable the corresponding outputs for binary mode.

Under *Config >> Device >> Output Mode* select the desired bnary access.

Required access rights: *Administrator*



Once you have made all your settings, send them to the Web-IO by clicking on the *Logout* button. Clicking on the *Save* button activates the settings.

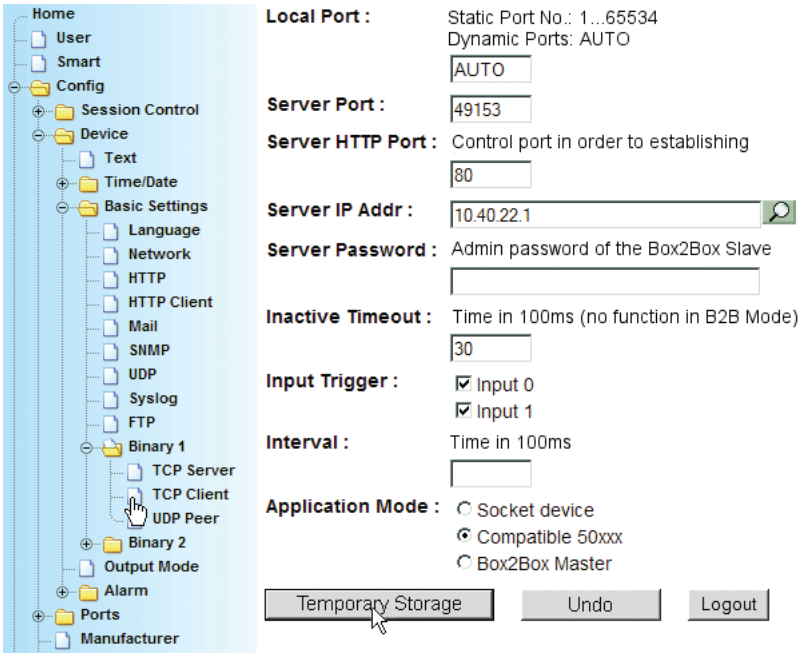
All the configuration possibilities shown for Binary 1 *can also be used for* Binary 2.

The Web-IO as socket client

To operate the Web-IO as a socket client, you must make a few additional settings.

Select in the navigation tree *Config >> Device >> Basic Settings >> Binary 1 >> TCP Client*

Required access rights: *Administrator*



Local Port

The local port of the Web-IO is factory set for 49153. If your application requires a different local port for the Web-IO, enter the desired port number in the *Local Port* field.

Server Port

Enter here the port number which the server application will use to accept the connection.

Server HTTP Port

Has meaning only in OPC and Box2Box modes and specifies the HTTP port on which the control connection to the OPC server or slave box should be opened.

Unless otherwised specified, always use Port 80 here.

Server IP Addr

Enter here the IP address of the server.

Server Password
A server password only needs to be entered if you want the Web-IO to work as a box-to-box master or access an additional Web-IO as a TCP client in server mode. See Box-to-Box section.

Inactive Timeout
Here you configure the timer used by the Web-IO to close the connection after the time has expired. Enter the value in decimal using 100ms increments. When there is an active network connection the timer is reset once all the data have been exchanged.

Example: The value 10 corresponds to one second. If no data transfer is detected during this time, the Web-IO closes the connection.

Entering no value means automatic connection closing is deactivated.

Input Trigger
Check the inputs whose state change you want to use as triggers for opening the TCP connection and sending data to the server (important for event-triggered applications).

Interval
If you want to cyclically notify the server application of input state changes, enter the interval here in (value will be multiplied with 100ms) .

Example: The value 300 corresponds to 30 seconds.

Please note that when using charged dial-up connections, too short an interval can result in this connection remaining open while charges continue to accumulate!

Mode
Select here:

- *Default* - If you want access to the Web-IO to be password protected.

- *Compatible 50xxx* - If you want to access applications that were programmed for the older Digital I/O Com-Servers. You can also use this mode for new applications that do not require password protection.

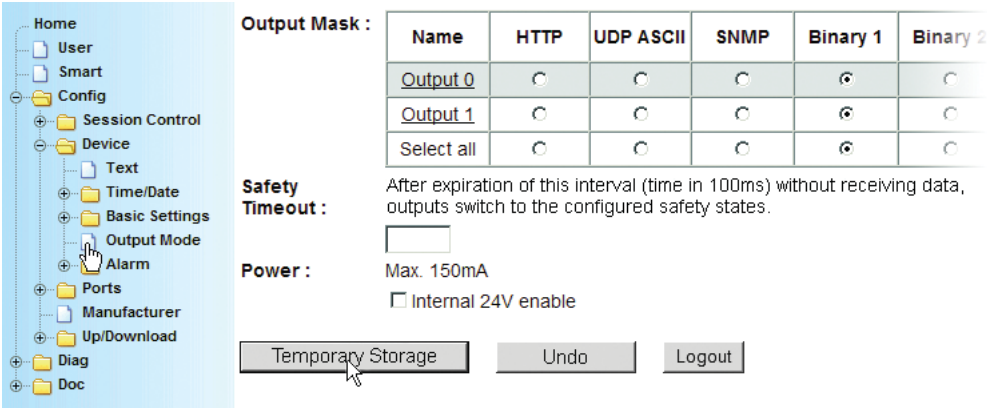
Additional information about *Box2Box Master* can be found in the section *Box-to-Box*.

Once you have made all your settings, send them to the Web-IO by clicking on *Temporary Storage*.

In addition you must enable the outputs you want to use for binary mode.

Under *Config >> Device >> Output Mode* check the desired binary access.

Required access rights: *Administrator*



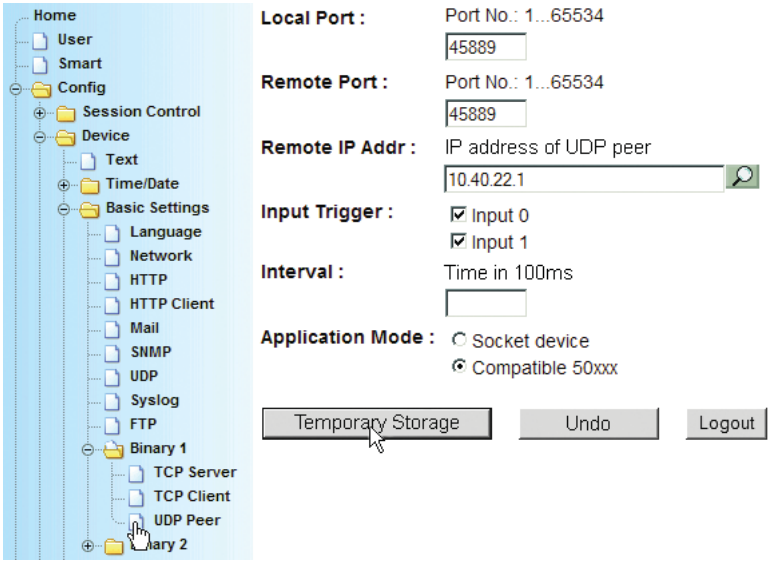
Once you have made all your entries, send them to the Web-IO by clicking on the *Temporary Storage* button.. Clicking on the *Save* button activates the settings.

The Web-IO as UDP Peer

To operate the Web-IO as a UDP peer, you must make a few additional settings.

Select in the navigation tree *Config >> Device >> Basic Settings >> Binary 1>> UDP Peer*

Required access rights: *Administrator*



Local Port
The local port of the Web-IO is factory set for 45889. If your application requires a different local port for the Web-IO, enter the desired port number in the *Local Port* field.

Remote Port
Enter here the port number the UDP application communicating with the Web-IO will use to receive data.

Remote IP Addr
Enter here the IP address of the communications partner.

Input Trigger
Check here the inputs whose state change will be used as a trigger for sending a UDP datagram (important for event-triggered applications).

Interval
If you want to send the state of the inputs cyclically to the communications partner, the interval can be entered here in 100ms increments.

Example: The value 300 corresponds to 30 seconds.

Please note that when using charged dial-up connections, too short an interval can result in this connection remaining open while charges continue to accumulate!

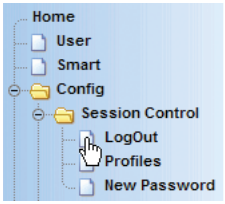
Application Mode
In UDP peer configuration, there is no difference between the Default and Compatible 50xxx modes.

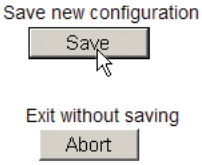
Once you have made all your settings, send them to the Web-IO by clicking on the *Temporary Storage* button.

In addition you must enable the corresponding inputs for binary mode.

Under *Config >> Device >> Output Mode* select the desired binary access.

Required access rights: *Administrator*





Once you have made all your entries, send the setting to the Web-IO by cliking on the *Logout* button. Clicking on the *Save* button activates the settings.

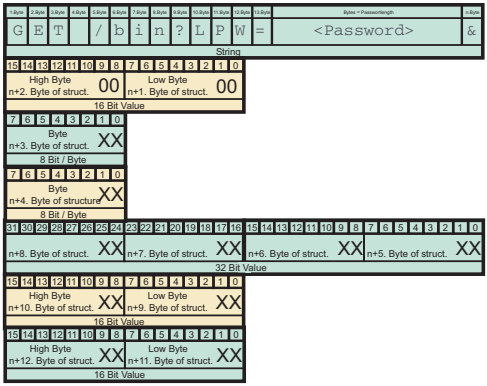
Passwort protection

As already mentioned several times, the Web-IO in TCP server mmode makes it possible to protect access to the application by means of a password..

Before the actual connection to the Web-IO is opened, the *BIN_INFO* structure defined here must be sent to Port 80 of the Web-IO over a separate TCP connection.

The Web-IO also uses the *BIN_INFO* structure for its reply.

The BinInfo structure


G E T / b i n ? L P W = <Password> &
String
High Byte n+2. Byte of struct. 00
Low Byte n+1. Byte of struct. 00
16 Bit Value
Byte n+3. Byte of struct. XX
8 Bit / Byte
Byte n+4. Byte of structure XX
8 Bit / Byte
n+8. Byte of struct. XX
n+7. Byte of struct. XX
n+6. Byte of struct. XX
n+5. Byte of struct. XX
32 Bit Value
High Byte n+10. Byte of struct. XX
Low Byte n+9. Byte of struct. XX
16 Bit Value
High Byte n+12. Byte of struct. XX
Low Byte n+11. Byte of struct. XX
16 Bit Value

HTTPLogin	String
Dummy	Word (16 Bit) always 0
Type	Byte (8 Bit) type of request
SubType	Byte (8 Bit) subtype of request
IPAddress	Longword (32 Bit) source IP address
SrcPort	Word (16 Bit) TCP source port
DstPort	Word (16 Bit) TCP destination port

The individual variables of the structure are filled as follows::

HTTPLogin[n]

Is a byte field or string consisting of a login string and the administrator password..

```
GET/bin?LPW=<AdministatorPasswort>&
```

n stands for the number of bytes used and corresponds to 14 + the length of the password. The length of the password is restricted to max. 31 characters.

In the reply from the Web-IO HTTPLogin is always 8 characters long and contains the following string::

```
GET/bin
```

Dummy
Separator between the ASCII and binary part of the structure..

is always = 0x00

Type
Determines the type in which the binary mode shall be used.

The application must enter 0x04 here to open a TCP connection.

In its reply the Web-IO enters

0x02 if the connection request was accepted.

0x03 if the connection request was rejected..

SubType
Gives additional details on the status of the connection request..

The application always sends 0x00.

The Web-IO replies with

```
0x01    BINSUBTYPE_OK,          // if the connection request was accepted..
0x02    BINSUBTYPE_NO_ACCESS,   // if a connection is already open
0x04    BINSUBTYPE_WAIT,        // If the connection is only allowed to be opened
                                // after a timeout
0x07    BINSUBTYPE_PW_MISMATCH,           // if wrong password
0x08    BINSUBTYPE_DEST_PORT_MISMATCH,    // if wrong Destination Port
0x09    BINSUBTYPE_MODUS_MISMATCH,        //ifwrongmode
```

If 0x01 or 0x04 was received, the actual data connection can be opened.

SrcPort
The client application always enters a 0 here.

The Web-IO returns here the opened server port (e.g., 49153 for binary 1). If the login attempt fails, the Web-IO enters 80.

DestPort
The client application enters here which port the connection is to be opened on. (e.g. 49153 for Binary 1 or 49154 for Binary 2)

The Web-IO always returns 0.

The connection through which the BinInfo structure was exchanged is automatically closed by the Web-IO.

BINARY - The IO structures
To enable simply communication between the application program on the computer and the Web-IO, there is a variety of structures (variable fields) which define the format and content of the data which are exchanged between the application program and the Web-IO.

IO structures are offered for the following functions:

- Reading inputs
- Reading inputs and outputs
- Setting outputs
- Reading counters
- Resetting counters
- Parameterizing cyclical and automatic messaging for state changes.

The application program employs the easy-to-use socket interface (Windows: WinSock, UNIX, Linux: Berkley Sockets) for exchanging data with the Web-IO over the network via TCP/IP in the form of these data structure.

The IO structures are independent of the selected network protocol (TCP or UDP).

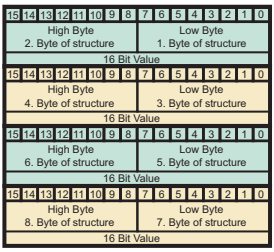
The choice of which protocol to use, UDP or TCP, depends on the type of application. Both protocols have their advantages and disadvantages that need to be weighed depending on the application you are creating.

Help for socket programming including the basics of TCP/IP can be found in a short, easy to understand guide „Ready for TCP/IP Sockets in 1 Day“. Program examples for client/server applications under TCP/IP can be downloaded from our homepage at http://www.wut.de.

Definition of the IO structures

To be able to unambiguously identify and evaluate the content of a packet, in BINARY mode all the data must be sent to the Web-IO in the form of these IO structures, regardless of whether *50xxx-compatible* or *Socket Device* mode is used.

All structures begin with the same header, which consists of the following 4 WORDS (16-bit_Integer):



Start_1	Word (16 Bit) always 0
Start_2	Word (16 Bit) always 0
StructType	Word (16 Bit) structure type
StructLength	Word (16 Bit) length in Bytes

Start_1, Start_2

Start_1 and *Start_2* are there for reasons of compatibility with earlier models, but they are not used. Both values are always 0.

StruktTyp

The value *struct_typ* identifies the structure. Both the PC application and the Web-IO use the value *struct_typ* when data are received to determine how the structure should be evaluated.

StructLength

length indicates the total length of the structure in bytes, i.e. including the first 4 WORDs.

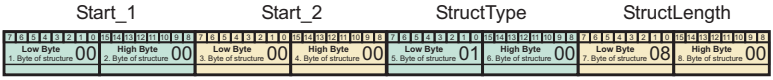
Note: The following applies to all IO structures:

A **word** corresponds to 16bit integer.

A **char** corresponds to one byte (8bits)
A **long** corresponds to a 32bit integer

Hexadecimal format **0x** in front of the value

For sending and receiving the variable **Low-Byte first** applies to all structure variables.


Start_1
Start_2
StructType
StructLength
Low Byte 1. Byte of structure 00
High Byte 2. Byte of structure 00
Low Byte 3. Byte of structure 00
High Byte 4. Byte of structure 00
Low Byte 5. Byte of structure 01
High Byte 6. Byte of structure 00
Low Byte 7. Byte of structure 08
High Byte 8. Byte of structure 00

In the examples all numbers are in hex format!

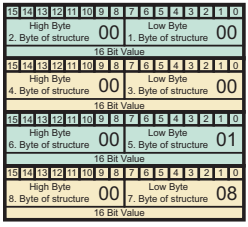
Working with the IO structures

In the next section the individual structures are explained and the corresponding values of the variables *Start_1*, *Start_2*, *struct_typ* and *length* which begin each packet are indicated.

The IO structure ReadRegister

Sending this structure to the Web-IO causes it to send the status of Inputs 0 - 11 to the application program. The packet consists only of these four WORDs. This structure is used by the user program and the Web-IO always responds by sending the WriteRegister structure.

The ReadRegister structure


High Byte 2. Byte of structure 00
Low Byte 1. Byte of structure 00
16 Bit Value
High Byte 4. Byte of structure 00
Low Byte 3. Byte of structure 00
16 Bit Value
High Byte 6. Byte of structure 00
Low Byte 5. Byte of structure 01
16 Bit Value
High Byte 8. Byte of structure 00
Low Byte 7. Byte of structure 08
16 Bit Value

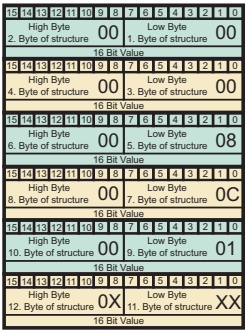
Start_1	Word (16 Bit) always 0
Start_2	Word (16 Bit) always 0
StructType	Word (16 Bit) structure type
StructLength	Word (16 Bit) length in Bytes

The IO structure WriteRegister

This structure is used to send the state of the inputs or outputs for the Web-IO 12xDigital. If the application program sends this structure to the Web-IO, the Web-IO sets the outputs corresponding to the value transmitted in *value*.

If the Web-IO sends this structure to the user program, *value* has the value corresponding to the input state.

The WriteRegister structure



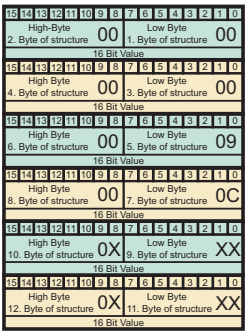
Start_1	Word (16 Bit) always 0
Start_2	Word (16 Bit) always 0
StructType	Word (16 Bit) structure type
StructLength	Word (16 Bit) length in Bytes
Amount	Word (16 Bit) always 0
Value	Word (16 Bit) binary input State

For Web-IOs with relay outputs you must maintain a pause of at least 200ms between two switching operations on one output!

The IO structure SetBit

This structure allows you to set individual outputs on the Web-IO 12xDigital. If for example the entire process status is not represented in the user program, individual output can be set without changing the value of the others. Bits 0..11 in *set_bits* and *value* correspond to the respective outputs. This structure is used only by the application program.

The SetBit structure



Start_1	Word (16 Bit) always 0
Start_2	Word (16 Bit) always 0
StructType	Word (16 Bit) structure type
StructLength	Word (16 Bit) length in Bytes
Mask	Word (16 Bit) binary output mask
Value	Word (16 Bit) binary output state

Example.:

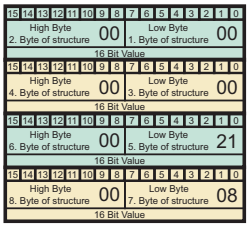
```
set_bits=0x0124/value=0x0104
```

Output 2 and Output 8 (counting Output0..11) are set to ON and Output 5 to OFF. No other outputs are changed.

The IO structure RegisterRequest

This structure sends the user program to the Web-IO 12xDigital so that the contents of inputs and outputs can be read in overview form. The Web-IO 12xDigital always responds with the I/O structure *RegisterState*

The RegisterRequest structure

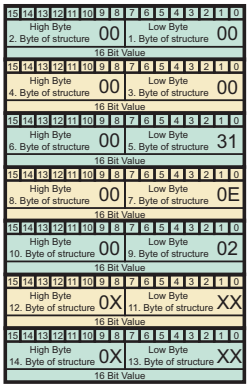


Start_1	Word (16 Bit) always 0
Start_2	Word (16 Bit) always 0
StructType	Word (16 Bit) structure type
StructLength	Word (16 Bit) length in Bytes

The IO structure RegisterState

The Web-IO 12xDigital sends the contents of the inputs and outputs with this structure. This structure is only sent if the user program has sent the structure *Register Request* to the Web-IO.

The RegisterState structure



Start_1	Word (16 Bit) always 0
Start_2	Word (16 Bit) always 0
StructType	Word (16 Bit) structure type
StructLength	Word (16 Bit) length in Bytes
DriverID	Word (16 Bit) Immer = 2
InputValue	Word (16 Bit) binary input state
OutputValue	Word (16 Bit) binary output state

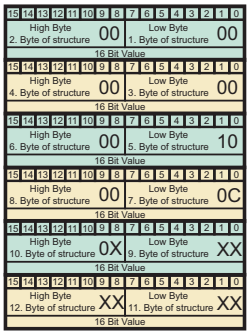
The IO structure Send Mode

This structure is used to specify the trigger conditions the Web-IO 12xDigital uses to send the status of the inputs to the user

program. There are basically three possibilities, but they may be combined with each other:

1. The user program polls the Web-IO by sending the READ structure
2. The Web-IO sends the WriteRegister - structure with the status of the inputs in a configurable interval
3. The Web-IO sends the WriteRegister - structure with the status of the inputs after a state change of the configured inputs

The SendMode structure

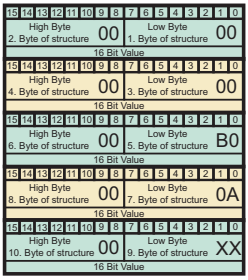

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
High Byte 2. Byte of structure 00
Low Byte 1. Byte of structure 00
16 Bit Value
High Byte 4. Byte of structure 00
Low Byte 3. Byte of structure 00
16 Bit Value
High Byte 6. Byte of structure 00
Low Byte 5. Byte of structure 10
16 Bit Value
High Byte 8. Byte of structure 00
Low Byte 7. Byte of structure 0C
16 Bit Value
High Byte 10. Byte of structure 0X
Low Byte 9. Byte of structure XX
16 Bit Value
High Byte 12. Byte of structure XX
Low Byte 11. Byte of structure XX
16 Bit Value

Start_1	Word (16 Bit) always 0
Start_2	Word (16 Bit) always 0
StructType	Word (16 Bit) structure type
StructLength	Word (16 Bit) length in Bytes
Mask	Word (16 Bit) binary input trigger mask
Interval	Word (16 Bit) sending interval fo input state

The IO structure ReadCounter

The user program sends this structure to the Web-IO to request the counter state of a certain input counter. Which input is intended is sent in the variable *counter_index*. The Web-IO always replies with the *COUNTER* structure.

The ReadCounter structure

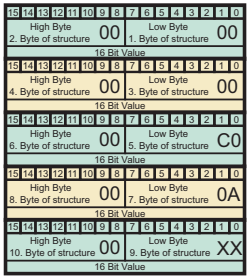

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
High Byte 2. Byte of structure 00
Low Byte 1. Byte of structure 00
16 Bit Value
High Byte 4. Byte of structure 00
Low Byte 3. Byte of structure 00
16 Bit Value
High Byte 6. Byte of structure 00
Low Byte 5. Byte of structure B0
16 Bit Value
High Byte 8. Byte of structure 00
Low Byte 7. Byte of structure 0A
16 Bit Value
High Byte 10. Byte of structure 00
Low Byte 9. Byte of structure XX
16 Bit Value

Start_1	Word always 0
Start_2	Word always 0
StructType	Word structure type
StructLength	Word length in Bytes
CounterIndex	Word counter index

The IO-Structure ReadClearCounter

The application program sends this structure to the Web-IO in order to request the counter status of a particular input counter and then immediately set to counter to 0. Which input this involves is transmitted in the variable *counter_index*. The Web-IO always replies with the structure *COUNTER*.

The ReadClearCounter structure

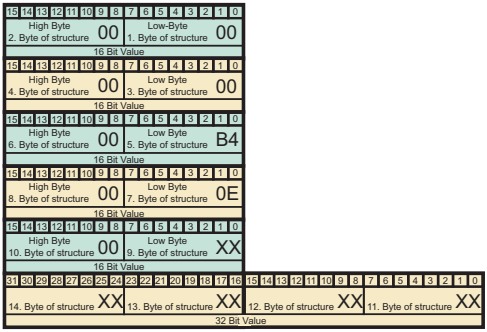


Start_1	Word always 0
Start_2	Word always 0
StructType	Word structure type
StructLength	Word length in Bytes
CounterIndex	Word counter indexl

The IO structure Counter

With this structure the Web-IO sends the counter state of the input counter specified in *counter_index*.

The Counter structure

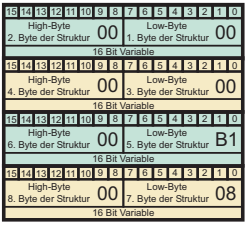


Start_1	Word always 0
Start_2	Word always 0
StructType	Word structure type
StructLength	Word length in Bytes
CounterIndex	Word counter index
CounterValue	Longword counter value

The IO structure ReadAllCounter

The user program sends this structure to the Web-IO to request the counter states of all the inputs in a data packet. The Web-IO always replies with the structure *ALL_COUNTER*.

Die ReadAllCounter Struktur

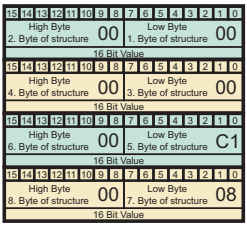


Start_1	Word Immer = 0
Start_2	Word Immer = 0
StructType	Word Strukturtype
StructLength	Word Länge in Bytes

The IO-Structure ReadClearAllCounter

The application program sends this structure to the Web-IO in order to request the counter states of all inputs in a data packet and then immediately sets the counters to 0. The Web-IO always replies with the structure *ALL_COUNTER*.

The ReadClearAllCounter structure

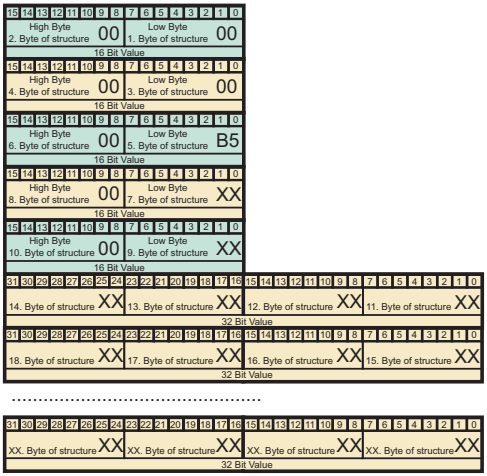


Start_1	Word always 0
Start_2	Word always 0
StructType	Word structure type
StructLength	Word length in Bytes

The IO structure AllCounter

The Web-IO uses this structure to send the counter states of all the inputs at one time.

The AllCounter structure

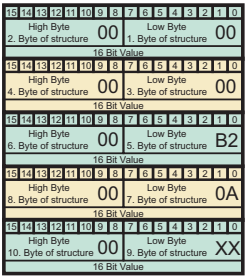

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
High Byte 2. Byte of structure 00
Low Byte 1. Byte of structure 00
16 Bit Value
High Byte 4. Byte of structure 00
Low Byte 3. Byte of structure 00
16 Bit Value
High Byte 6. Byte of structure 00
Low Byte 5. Byte of structure B5
16 Bit Value
High Byte 8. Byte of structure 00
Low Byte 7. Byte of structure XX
16 Bit Value
High Byte 10. Byte of structure 00
Low Byte 9. Byte of structure XX
16 Bit Value
31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
14. Byte of structure XX
13. Byte of structure XX
12. Byte of structure XX
11. Byte of structure XX
32 Bit Value
18. Byte of structure XX
17. Byte of structure XX
16. Byte of structure XX
15. Byte of structure XX
32 Bit Value
XX. Byte of structure XX
XX. Byte of structure XX
XX. Byte of structure XX
XX. Byte of structure XX
32 Bit Value

Start_1	Word always 0
Start_2	Word always 0
StructType	Word structure type
StructLength	Word length in Bytes
CounterNoOf	Word number of counters
CounterValue1	Longword counter value 1
CounterValue2	Longword counter value 2
CounterValuen	Longword counter value n

The IO structure ClearCounter

The user program sends this structure to the Web-IO to reset the counter state of a certain input counter to 0. Which input is intended is sent in the variable *counter_index*

The ClearCounter structure

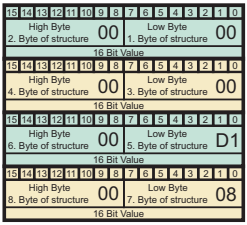

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
High Byte 2. Byte of structure 00
Low Byte 1. Byte of structure 00
16 Bit Value
High Byte 4. Byte of structure 00
Low Byte 3. Byte of structure 00
16 Bit Value
High Byte 6. Byte of structure 00
Low Byte 5. Byte of structure B2
16 Bit Value
High Byte 8. Byte of structure 00
Low Byte 7. Byte of structure 0A
16 Bit Value
High Byte 10. Byte of structure 00
Low Byte 9. Byte of structure XX
16 Bit Value

Start_1	Word always 0
Start_2	Word always 0
StructType	Word structure type
StructLength	Word length in Bytes
CounterIndex	Word counter index

IO-Structure ReadDiagnosis

If the Web-IO determines that there is a communications or system error, the latter is listed on the HTML page *diag* and can be read using the browser. Since error management via browser for program-controlled applications is not always available, the error status of the Web-IO can be queried using the structure *ReadDiagnosis*

The ReadDiagnosis structure


15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
High Byte
2. Byte of structure 00
Low Byte
1. Byte of structure 00
16 Bit Value
High Byte
4. Byte of structure 00
Low Byte
3. Byte of structure 00
16 Bit Value
High Byte
6. Byte of structure 00
Low Byte
5. Byte of structure D1
16 Bit Value
High Byte
8. Byte of structure 00
Low Byte
7. Byte of structure 08
16 Bit Value

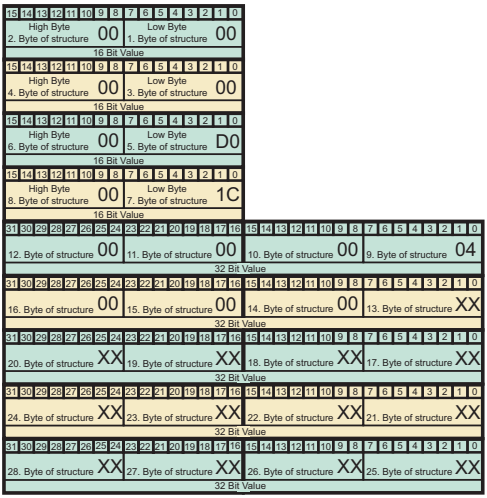
Start_1	Word (16 Bit) always 0
Start_2	Word (16 Bit) always 0
StructType	Word (16 Bit) structure type
StructLength	Word (16 Bit) length in Bytes

The Web-IO replies with a Diagnosis type structure.

IO-Structure Diagnosis

The Web-IO replies to the *ReadDiagnosis* structure with a Diagnosis type structure.

The Diagnosis structure


15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
High Byte
2. Byte of structure 00
Low Byte
1. Byte of structure 00
16 Bit Value
High Byte
4. Byte of structure 00
Low Byte
3. Byte of structure 00
16 Bit Value
High Byte
6. Byte of structure 00
Low Byte
5. Byte of structure D0
16 Bit Value
High Byte
8. Byte of structure 00
Low Byte
7. Byte of structure 1C
16 Bit Value
31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
12. Byte of structure 00
11. Byte of structure 00
10. Byte of structure 00
9. Byte of structure 04
32 Bit Value
16. Byte of structure 00
15. Byte of structure 00
14. Byte of structure 00
13. Byte of structure XX
32 Bit Value
20. Byte of structure XX
19. Byte of structure XX
18. Byte of structure XX
17. Byte of structure XX
32 Bit Value
24. Byte of structure XX
23. Byte of structure XX
22. Byte of structure XX
21. Byte of structure XX
32 Bit Value
28. Byte of structure XX
27. Byte of structure XX
26. Byte of structure XX
25. Byte of structure XX
32 Bit Value

Start_1	Word (16 Bit) always 0
Start_2	Word (16 Bit) always 0
StructType	Word (16 Bit) structure type
StructLength	Word (16 Bit) length in Bytes
Version	Longword (32 Bit) in this release V4
ErrorCount	Longword (32 Bit) number of errors
ErrorBits1	Longword (32 Bit) binary error coding
ErrorBits2	Longword (32 Bit) binary error coding
ErrorBits3	Longword (32 Bit) binary error coding

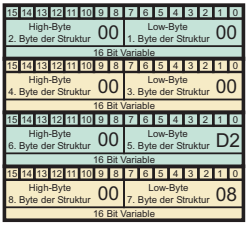
The variable diag_error_count returns how many various errors are currently in the error memory. The Web-IO distinguishes a total of up to 64 different error states, whereby each set bit of the two variables diag_errrorbits0 and diag_errorbits1 stands for an error type.

The exact text description can be obtained on TCP Port 80. For a more detailed description, see the section on socket programming with command strings.

IO-Structure ClearDiagnosis

This structure clears the error memory of the Web-IO.

Die ClearDiagnosis Struktur

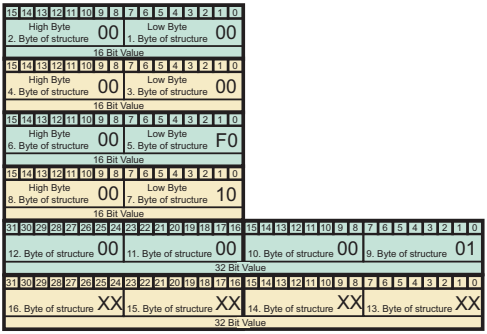

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
High-Byte 2. Byte der Struktur 00
Low-Byte 1. Byte der Struktur 00
16 Bit Variable
High-Byte 4. Byte der Struktur 00
Low-Byte 3. Byte der Struktur 00
16 Bit Variable
High-Byte 6. Byte der Struktur 00
Low-Byte 5. Byte der Struktur D2
16 Bit Variable
High-Byte 8. Byte der Struktur 00
Low-Byte 7. Byte der Struktur 08
16 Bit Variable

Start_1	Word (16 Bit) Immer = 0
Start_2	Word (16 Bit) Immer = 0
StructType	Word (16 Bit) Strukturtype
StructLength	Word (16 Bit) Länge in Bytes

IO-Structure Options

This structure is used to set certain options in the Web-IO. The *options* variable provides 32 bits for this.

The Options structure


15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
High Byte 2. Byte of structure 00
Low Byte 1. Byte of structure 00
16 Bit Value
High Byte 4. Byte of structure 00
Low Byte 3. Byte of structure 00
16 Bit Value
High Byte 6. Byte of structure 00
Low Byte 5. Byte of structure F0
16 Bit Value
High Byte 8. Byte of structure 00
Low Byte 7. Byte of structure 10
16 Bit Value
31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16
12. Byte of structure 00
11. Byte of structure 00
10. Byte of structure 00
9. Byte of structure 01
32 Bit Value
16. Byte of structure XX
15. Byte of structure XX
14. Byte of structure XX
13. Byte of structure XX
32 Bit Value

Start_1	Word (16 Bit) always 0
Start_2	Word (16 Bit) always 0
StructType	Word (16 Bit) structure type
StructLength	Word (16 Bit) length in Bytes
Version	Longword (32 Bit) in this release V4
Options	Longword (32 Bit) binary option coding

In the current version of the Web-IO only Bit 0 in the *options* variable is used.

Bit 0 = 1 //the Web-IO returns the Structure register state when an output is set.
Bit 0 = 0 //the Web-IO does not reply when setting an output.

To ensure that the state after setting the output(s) is reliably returned, there should be a time of 150ms between two output-changing accesses.

5.7 Wayback - switching via FTP login

In addition to the standard modes, the ERP gateway models 57635, 57638 and 57636 also include the patented Wayback procedure for accessing the digital outputs. This makes it possible to send switching commands using the login name of an FTP client. In addition, communicatively limited systems such as SAP or other ERP/PPS applications are made able to directly access the in-/outputs - without any special drivers, special software packages or particular user access rights to the file system.

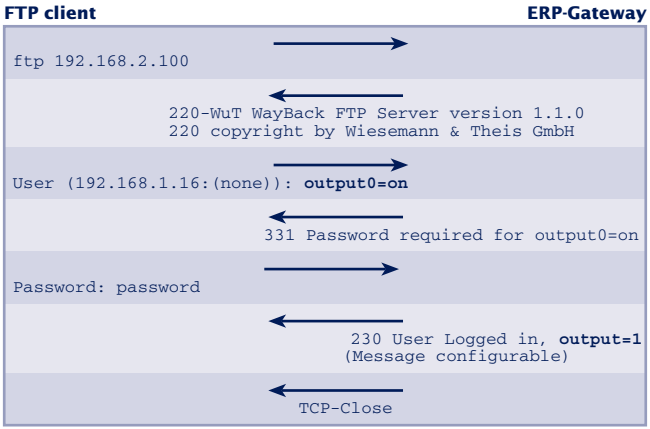
In contrast to the other Web-IO Digital versions, models 57635, 57638 and 57636 have the Wayback FTP preconfigured as a factory default setting.

5.7.1 Functional principle

After the TCP connection is opened, the Wayback FTP server sends its welcome message to the FTP client. Now the command for switching the outputs or reading the outputs is entered as the *User Name*. As soon as the command is authenticated using the *Wayback Password* the command is executed and for example the specified output is switched to the desired state. The Wayback server sends the configurable terminating message and then automatically closes the connection.

To prevent simultaneous, competing access attempts, the Wayback service is designed to be exclusive. This means that after a client opens the connection, this connection must first be closed by a valid command/password combination or by the timeout before a new access is accepted.

Sequence example of a switching command for Output 0:



5.7.2 The Wayback commands

The following commands can be sent as an FTP User Name to the ERP gateway. The commands are not case sensitive. All replies from the ERP gateway are send using the variable *<MSG>* in the *Wayback Response*.

output[*x*]=[*on*|*off*]

x Number of the desired output, beginning with 0 ro the first output of an ERP gateway
on Switches Output *x* on
off Switches Output *x* off

The command enables targeted on-/off-switching of individual ERP gateway outputs.

Example:
output0=on turns Output 0 on the ERP gateway ON

output=[*xxxx*]

xxxx Bit pattern of the desired state of all outputs in hex format.

Example:
output=2 (binary = 0000 0010) turns Output 1 ON, and all other available outputs are turned OFF

output=3e (binary = 0011 1110) turns Outputs 1 to 5 ON, and all other available outputs are turned OFF.

output

This command enables reading of the state of the digital outputs. The reading is a hex value for the binary state of all outputs using the variable *<MSG>* in the *Wayback Response.*

input

This command enables reading of the state of the digital inputs. The reading is a hex value for the binary state of all inputs using the variable *<MSG>* in the *Wayback Response.*

counter[*x*]

x Number of the desired input beginning with 0 for the first output of the ERP gateway

The command allows you to read the counter of the specified digital input. The value is sent as a decimal value with the variable *<MSG>* in the *Wayback Response.*

5.7.3 Enabling Wayback access

Required access rights: *Administrator*

In the navigation tree for the ERP gateway select *Config >> Device >> Basic Settings >> WayBack FTP*

You can make the following settings on this page:

Enable

Activates the Wayback service. If this option is deactivated, the Wayback server service is not started and connection attempts to the *FTP Control Port* are rejected.

FTP Control Port

Indicates the TCP server port at which the Wayback service is accessible for FTP clients. The port reserved for FTP and used by most FTP clients as the standard is TCP port 21.

Password

The Wayback password entered on the FTP client for correct execution of the command. This is *not* identical to the *Admin* or *Operator* password.

Wayback Response

Final message sent to the FTP client after receipt of the correct Wayback password. After this message the EROP gateway automatically closes the TCP conneciton. The following variables may be used within the text:

<MSG>

The ERP gateway replaces this variable with the status of the outputs corresponding to the previous comnmand. The information is sent as a hex equivalent of the binary state of all in-/outputs or as a decimal value for the counters. If an output command was sent and Outputs 0 and 1 are both ON, the response is for example *output=3*. If Outputs 0, 4 and 5 are ON on a model 57636 with 6 relay outputs, *output=31* is returned.

<CRLF>
The ERP gateway repalces this variable with the characters CR (Carriage Return, 0x0d) and LF (Linefeed, 0x0a). LIne breaks inserted in the input window using the ENTER key are *not* included in the reply.

Connect Timeout

After the specified time in seconds has expired with no activity form the FTP client, the ERP gateway closes the existing TCP connection.

After configuring the desired parameters, the changes are saved by clicking on the *Save* button.

If no other configuration changes are required, you must still activate the new settings. Use the *Logout* button or menu path *Config >> Session Control >> LogOut*

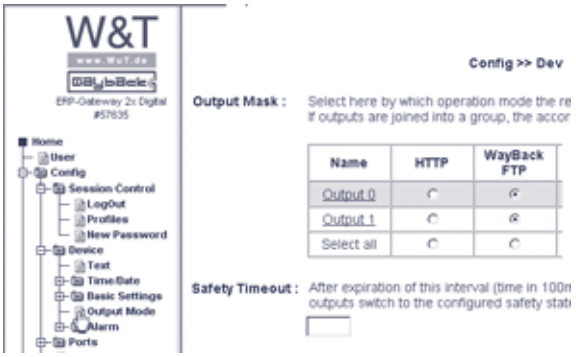
5.7.4 Activating the outputs for Wayback access

Required access rights: *Administrator*

The ERP gateways 57635 and 57636 are factory configured for using the Wayback procedure, so that

when first starting up the unit you do not need to activate the outputs as shown here.

In the ERP gateway navigation tree select *Config >> Device >> Output Mode*



Here you activate the desired outputs you want to control using Wayback and then click on the *Save* button.

If no other configuration changes are required, you must still activate the new settings. Use the *Logout* button or menu path *Config >> Session Control >> LogOut.*

6 General Functions

- Time / Date
- Poll watchdog - access monitoring
- Port configuration
- Changing the device-specific displays
- Upload / Download
- Diagnostics and testing

6.1 Time / Date - Setting

The Web-IO Digital has an internal clock with date functions.

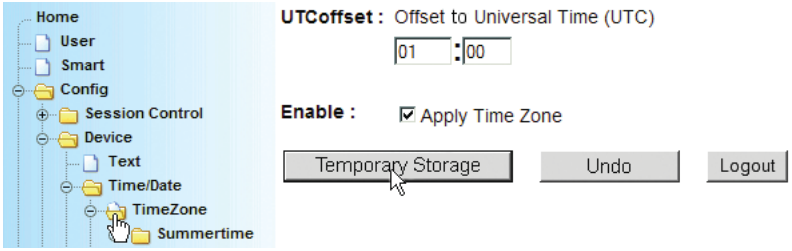
6.1.1 Time zones

Required access rights: *Administrator*

The Web-IO usesUTC time (universal time coordinated) internally, which is used as a general time base all over the World. The local time deviates from UTC depending on the time zone you are in. The time in Germany is UTC + 1 h (in winter), the time in the United States is UTC - 5h, etc.

If the Web-IO will be used in another time zone, the offset can be adjusted accordingly.

To do this, go to *Config >> Device >> Time/Date >> TimeZone:*



UTCoffset
Enter here the difference between local and UTC time in the format hh.mm. A sign can also be used in the entry field („+“ or „-“).

Enable
To activate, check Apply Timezone to use the entered offset value.

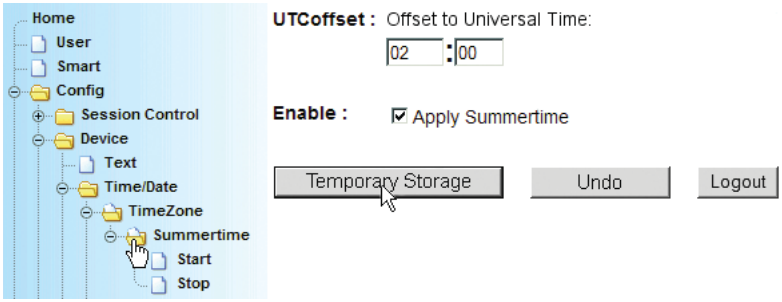
Send the settings to the Web-IO by clicking on the *Temporary Storage* button.

6.1.2 Summer time / Winter time

Required access rights: *Administrator*

The Web-IO automatically distinguishes between summer and winter time.

To change the reference values for summer/winter time, select in the navigation tree *Config >> Device >> Time/Date >> TimeZone >> Summertime:*



UTCOffset
Enter here the difference between local summer time and UTC time in the format hh.mm. A sign can also be used in the entry field („+“ or „-“).

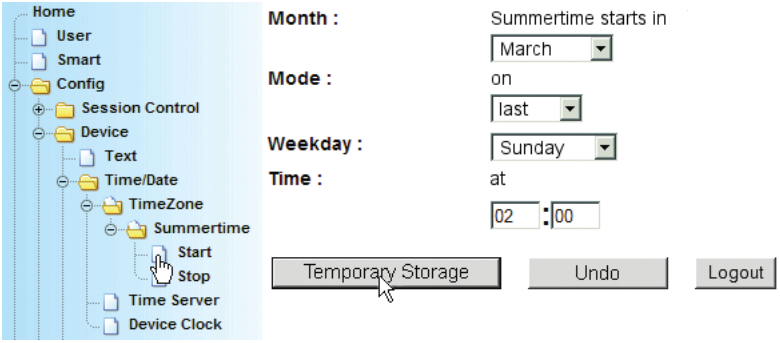
Enable
To activate, check *Apply Summertime* to use the entered offset value.

Send the settings to the Web-IO by clicking on the *Temporary Storage* button.

6.1.3 Start and end of summer time

Required access rights: *Administrator*

To set when summer time begins, select *Config >> Device >> Time/Date >> TimeZone >> Summertime >> Start:*



Month
Select the month in which the change to summer time takes place.

Mode
If summer time does not begin on the last Sunday of the selected month, you can select a different mode here.

Weekday
If the time change does not take place for example from Saturday to Sunday, you can select a different day of the week.

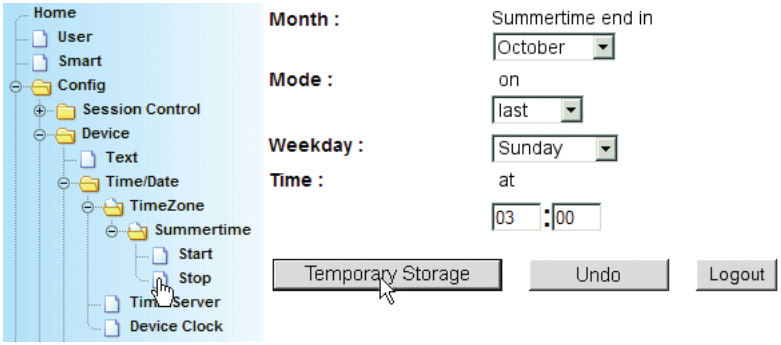
Time
Time at which the change takes place.

Send the settings to the Web-IO by clicking on the *Temporary Storage* button.

6.1.4 End of summer time

m Required access rights: *Administrator*

To select when summer time ends, select *Config >> Device >> Time/Date >> TimeZone >> Summertime >> Stop:*



Month
Select the month in which summer time ends.

Mode
If summer time does not change on the last Sunday of the selected month, you can select a different mode here.

Weekday:
If the time change does not take place for example from Saturday to Sunday, you can select a different day of the week.

Time
Time at which the change takes place.

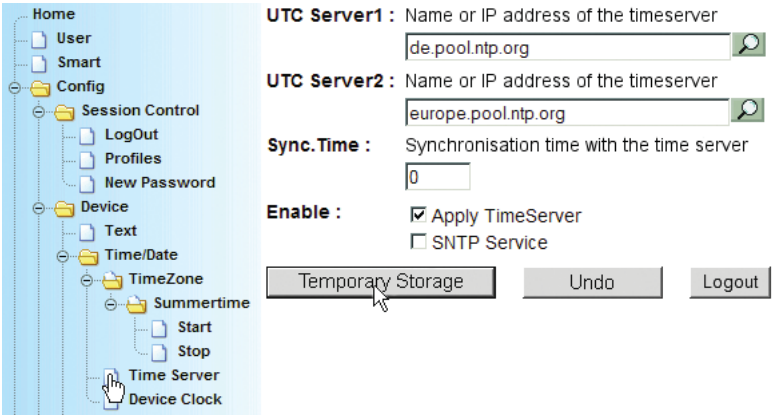
Send the settings to the Web-IO by clicking on the *Temporary Storage* button.

6.1.5 Time-Server

Required access rights: *Administrator*

A time server can be used to synchronize the internal clock of the Web-IO. The Web-IO checks against the time server after it is powered up. There is also a daily time synchronisation at a selected time every day.

To enter the necessary settings for the time server, select in the navigation tree *Config >> Device >> Time/Date >> Time Server:*



UTC-Servers 1 + 2
Enter here one or two time servers that can be reached from the location of the Web-IO.

Sync Time
The hour the time will be synchronized with an external time server.

Synchronization with the time server takes also place approx. 30 seconds after the Web-IO is turned on or after a reset. If the Web page of the Web-IO is opened immediately after a new start, the old unsynchronized time will still be displayed. Only after opening the page again will the current time of day be shown.

Enable
Select *Apply Timeserver* so that the time of day will from now on be synchronized with an external time server.

Send the settings to the Web-IO by clicking on the *Temporary Storage* button.

The factory default addresses can only be reached via an Internet connection. If names instead of IP addresses

were specified for the time servers, a DNS server must be specified in the network configuration.

6.1.6 Manually setting time and Date

Required access rights: *Administrator* oder *Operator*

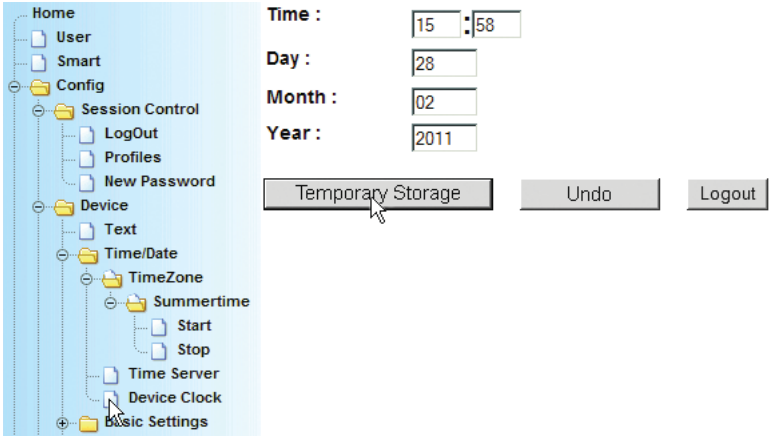
If there is no Internet access to a time server, the time of day can also be set manually under *Config >> Device >> Time/Date >> Device Clock.*

Time
Time in format hh:mm. It is not necessary to set to second accuracy.

Day
Format dd

Month
Format mm

Year
Format yyyy



Send the settings to the Web-IO by clicking on the *Temporary Storage* button.

6.1.7 Activating all time/date settings

Once all the entries pertaining to the time and date have been made, activate the new setting under *Config >> Session Control >> Logout*, and then click the *Save* button.

If you log in again as a Configurator or Administrator immediately after activating the time server synchronization, it can happen that access will be denied to you after a few seconds.

This is because for security reasons the Web-IO resets the login rights after one hour. If for example the Web-IO has a time setting of 12:00 hours on 01.01.2002 (factory default at power on) just when the login takes place and the time server sets the time to 13:42 hours on 26.06.2003, the time span is considerably greater than one hour and the Web-IO cancels the access rights.

After logging in again the full hour can then be fully used up for configuration.

6.2 Safety Timeout / State - Access monitoring

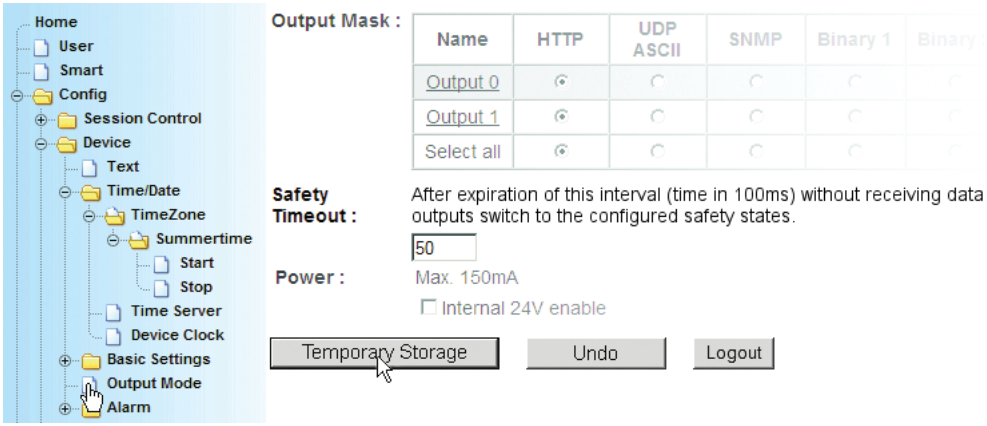
Components that communicate with each other over a network must rely on a functioning network infrastructure. Even though today's networks offer a high degree of transmission security, problems can never be entirely precluded.

To ensure predictable behavior of the Web-IO when there is a network disturbance, a *Safety Timeout* is provided. The associated watchdog monitors whether there has been a network access to the Web-IO within a configurable interval of time.The watchdog timer is reset each time there is a network access and plces the outputs in the configurable state when the set *Safety Timeout* time has been reached.

6.2.1 Specifying safety timeout

To configure the safety timeout, select in the navigation tree *Config >> Device >> Output Mode:*

Required access rights: *Administrator*



Safety Timeout
Enter here the time that should elapse before the absence of a network access causes the outputs to be set to the set pattern.

The entered value times 100ms is the timeout time; you can enter values between 1 and 9999.

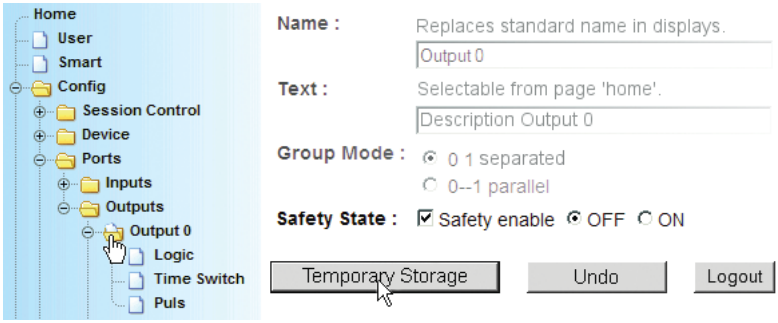
Example: a value of 50 represents 5 seconds.

As a rule of thumb, use a timeout that is 10x longer than the time between two scheduled or expected accesses.

Send the settings to the Web-IO by clicking on the *Temporary Storage* button.

6.2.2 Specify Safety State

Select in the navigation tree *Config >> Ports >> Outputs >>* and for example *Output 0*



Safety State
Select *Safety State Enable* if you want the output to be set to a fixed state when there is a fault. Specify whether the output shall be switched to the *ON* or *OFF* state.

Outputs that are not selected keep the state that was present when the fault occurred.

All other settings on the page are not relevant to the *Safety State* and do not have to be changed.

Send the settings to the Web-IO by clicking on the *Temporary Storage* button.

Once you have made all the settings, activate them under *Config >> Session Control >> LogOut*, *Save.*

6.3 Ports - Input- and output configuration

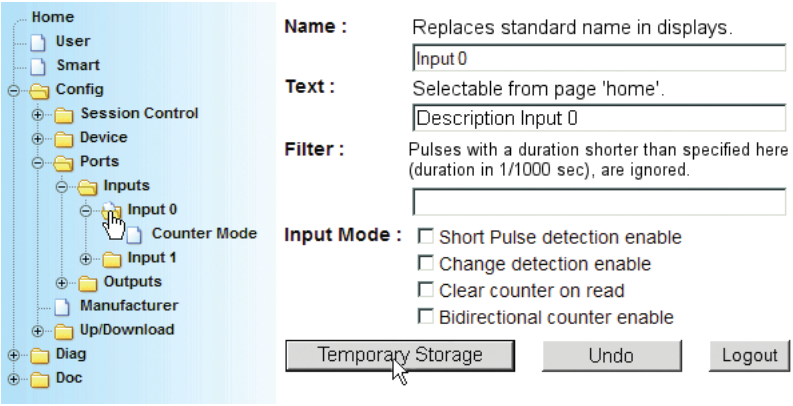
6.3.1 Input configuration

The inputs are organized into groups of 4. In addition to the galvanic isolation of the groups, Group 1 has the unique feature that the inputs can be selected for edge or pulse detection.

Individual base settings can be made for each input.

For example, to change the settings for Input 0, select in the navigation tree *Config >> Ports >> Inputs* and for Input 0 *>> Input 0*:

Required access rights: *Administrator* or *Config*



Name
Enter here a name for the input. This name will then be displayed in the browser for Input 0.

Text
Here you can provide a more detailed description, including for example the function of the input, which can then be viewed by clicking on the Input link in the browser.

Filter
Pulses with a duration shorter than specified here (duration in1/1000sec),are ignored.

Whenusing inputs that work with Short Pulse Detection, the configured filters have no effect

Input Mode: Short Pulse Detection Enable
This mode is available only for inputs 0 - 3 and also allows detection of extremely short pulses down to 50µs in length. In this mode only complete pulses, not edge transitions, can be counted.

If *Pulse Detection* is not enabled, signals must be present for at least 1ms in order to be detected and counted.

In both cases a maximum of 1000 counts per second can be detected.

Input Mode: Pulse Detection
Select this field if you want to configure a pulse count for the input.

This setting affects the way the counter counts.

If *Pulse Detection* is not activated each state change on the inputs is counted. A change from OFF to ON increments the counter just as a change from ON to OFF. So a complete puls will increment the counter by two. In this mode, has to be one state present for at least 1ms to be recorded.

If Pulse Detection is activated, there must be a change from ON to OFF and then a change from OFF to ON before the counter is incremented. The ON state must be present for at least 50ms for the pulse to be detected and counted.

In both cases, a maximum of 1000 state changes per second can be counted.

Input Mode: Change Detection Enable
In addition to counting complete pulses, here you can switch to counting edge changes.

This has no meaning if Short Puls Detection enable is active.

Input Mode: Clear counter on read
Here you can specify whether the counter is automatically set to 0 when it is read.

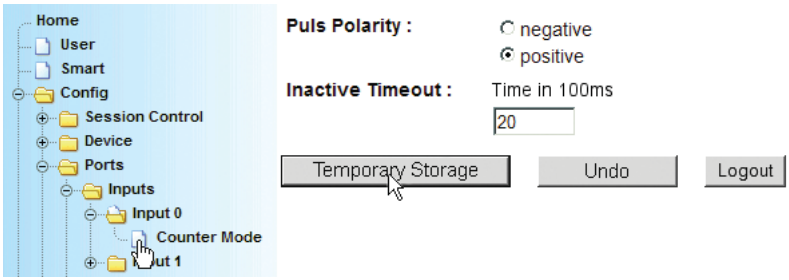
Input Mode: Bidirectional counter enable
Two adjacent inputs are combined for processing direction-dependent encoders.

Depending on which of the two inputs a valid edge is first detected, the corresponding counter is incremented by one.

Send the settings to the Web-IO by clicking on the *Temporary Storage* button.

The necessary settings for direction-dependent counting are made under *Config >> Ports >> Inputs >> Input0 >> Counter Mode*.

Required access rights: *Administrator*



Pulse Polarity
Select here whether you want the Web-IO to trigger on positive or negative edges

Inactivity Timeout
If no valid edges are detected on either of the inputs within the set time, the counting is cancelled.

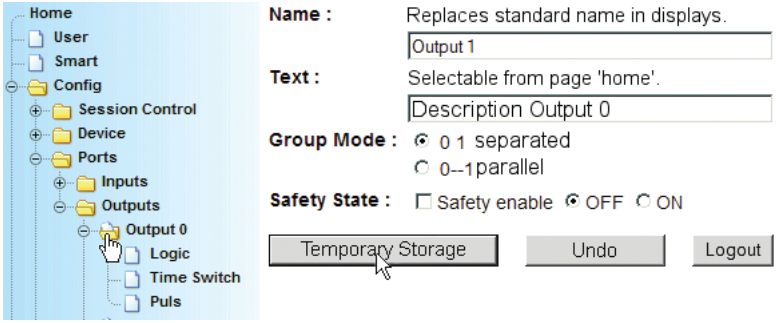
Send the settings to the Web-IO by clicking on the *Intermediate store* button.

Once you have made all the entries, activate the new settings under *Config >> Session Control >> LogOut, Save.*

6.3.2 Output configuration

To change the settings for Output 0 for example, select *Config >> Ports >> Outputs* and *>> Output 0:*

Required access rights: *Administrator* oder *Config*



Name
Enter here a name for the input. This name will then be displayed in the browser for Output 0.

Text
Here you can provide a more detailed description, including for example the function of the output, which can then be viewed by clicking on the Output link in the browser.

Group Mode
The outputs can be combined into groups of two or four to increase the maximum switching current. Combining two

outputs allows you to switch a maximum current of 1A. With four outputs this increases to 2A. *(not applicable to Web-IO with relay outputs)*

Safety State
Select *Safety State Enable* if you want the output to be set to a fixed state when there is a fault. Specify whether the output shall be switched to the *ON* or *OFF* state.

This will only have an effect if a time is selected for *Safety Time-out* in *Config >> Device >> Output Mode*

A detailed description of *Safety State* can be found in the section *Safety Timeout / State.*

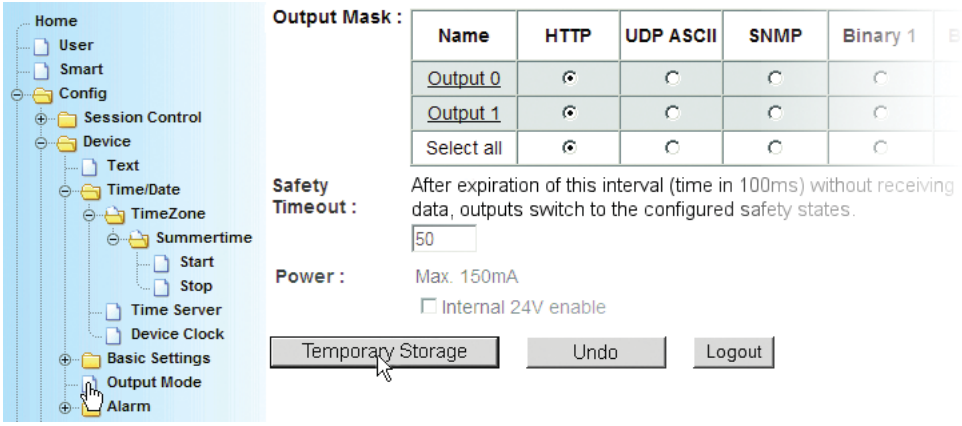
Send the settings to the Web-IO by clicking on the *Temporary Storage* button.

Once you have made all the entries pertaining to the alarm, activate the new settings under *Config >> Session Control >> LogOut, Save.*

Outputs that are combined in a group should be configured for the same operation mode. Otherwise conflicts can arise if for example Outputs 0 and 1 are in Box-to-Box nmode and Outputs 2 and 3 can be switched from the browser, since each access switches all the outputs in a group at the same time.

An overview of the configured output groups can be found under *Config >> Device >> Output Mode.*

Required access rights: *Administrator*

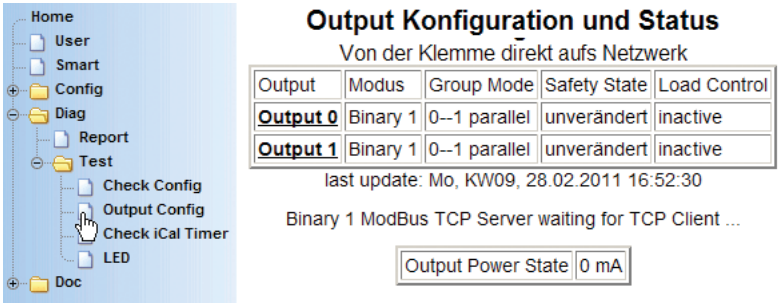


Here the outputs that are combined in a group are highlighted in the same color.

In the example shown, Outputs 0 - 3 represent a group. Outputs 4 and 5 and Outputs 6 and 7 represent two additional groups. Outputs 8,9,10 and 11 are ungrouped, and can be switched individually.

An additional way to see the groups at a glance is under *Diag >> Test >> Output Config* in the column *Group Mode.*

Required access rights: *Administrator*



6.3.3 Logic functions

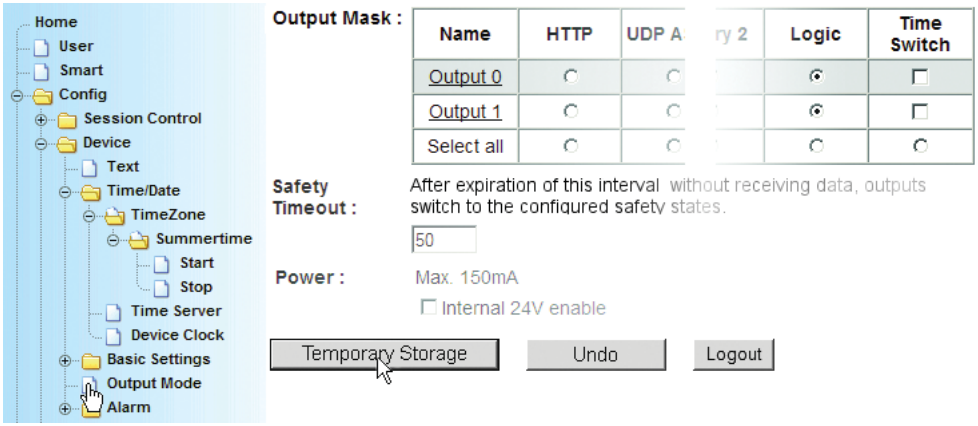
The Web-IO allows outputs to be switched depending on the state of certain inputs.

Example
Output 0 is set to ON when Input 0 and Input 1 are OFF.

For the logic functions to be applied to an output, the latter must first be configured to *Logic* mode.

To do this, use the path *Config >> Device >> Outputmode.*

Required access rights: *Administrator*

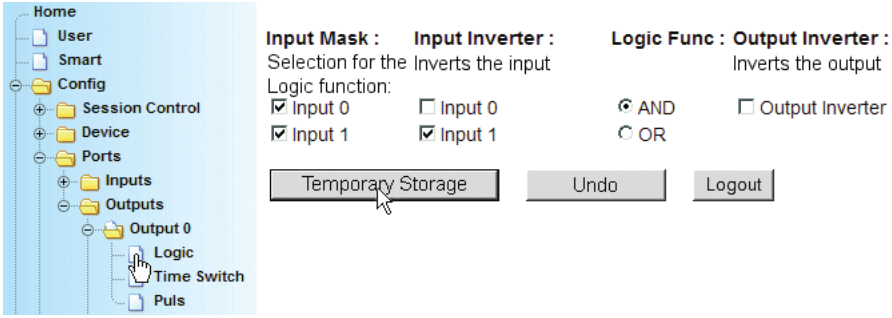


Click on the *Logic* column for the outputs you want to control using the Logic function and then click on *Save*.

Select in the navigation tree *Config >> Ports >> Outputs >> Output 0 >> Logic:*

Required access rights: *Administrator* or *Config*

The example above looks like this in the browser:



Input Mask
Select here all the inputs you want to be included in the logic operation.

Input Inverter
Here you specify whether the state of the input should be inverted..

☐ = Input is evaluated as it is
☑ = The inverse of the input state is evvaluated

Logic Func
Specifies whether an AND or OR operation is applied.

Output Inverter
Here you specify whether fulfillment of the logic conditions sets the output to ON or OFF.

☐ = Output is set to ON
☑ = Output is set to OFF

Example
The configuration illustrated sets Output 0 to ON when Input 0 is ON and Input 1 is OFF.

Send the settings to the Web-IO by clicking on the *Temporary Storage* button.

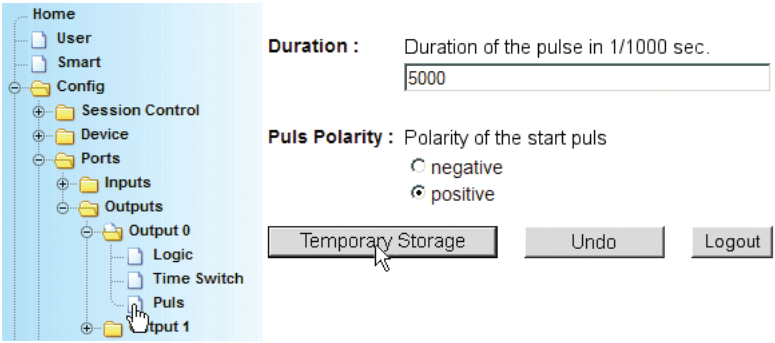
Once you have entered the conditions for all the outputs, activate the new settings under *Config >> Session Control >> LogOut, Save.*

6.3.4 Pulse mode

In addition to purely static switching of the outputs to ON or OFF, the Web-IO also permits pulses to be output. This means an output can be switched to On or OFF for a preset amount of time and resumes its base state after the set pulse length.

To configure the Web-IO for outputting pulses, select Config >> Ports >> Pulse

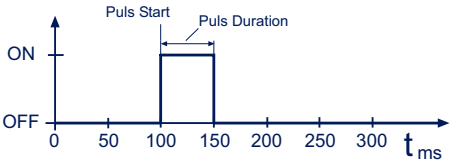
Required access rights: *Administrator*



Duration
Enter here the desired pulse length in ms. A value of 1000 corresponds to a 1-second long pulse.

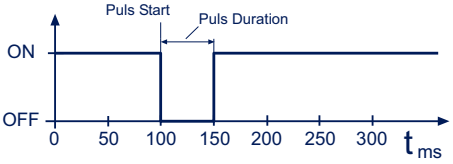
Normally an output can not be switched back to the rest state by the user during the pulse. If that is desired, a capital *R* can be inserted behind the pulse length (e.g. 2000R).

Pulse Polarity
positive:



In the base state the output is not switched (0V). If the output is now set to ON via browser or from an application, the Web-IO switches the supply voltage +Vdd to the output for the set pulse duration.

negative:



In the base state the output is switched to supply voltage +Vdd. If the output is now set to ON from an application, the Web-IO turns the output off (0V) for the set pusle duration.

If *0* is entered as a pulse length for *Pulse Polarity negative*, the physical output signal is inverted.

For the Web-IO with relay outputs a pause of min. 200ms must be maintained between two switching operations on an output!

With negative polarity negative logic is used. Switching to ON causes the corresponding output to be turned off.

STOP *It is not possible to switch a negative pulse via HTTP from the browser. If you need to work with negative pulses from the browser, this can be done using the Java applet.*

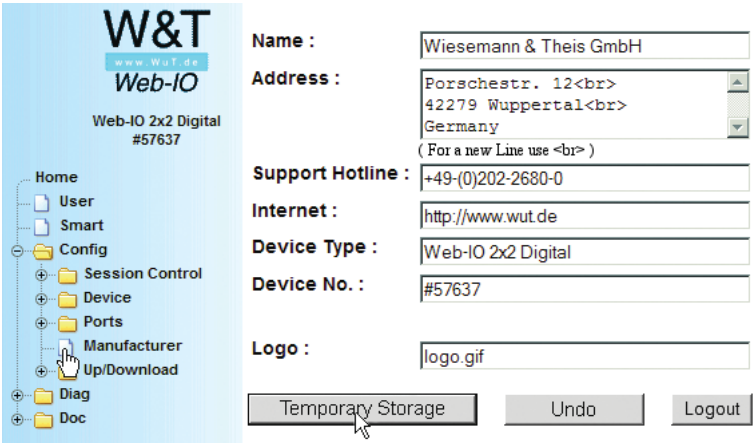
Logout and *Save* activates the new setting.

6.5 Changing device-specific displays

For project-specific OEM applications of the Web-IO you can change displays such as article number, manufacturer, logo, etc.

Select in the navigation tree *Config >> Manufacturer:*

Required access rights: *Administrator*



Name
Field for the company name

Address
Field for the company address

Support Hotline
Telephone number the user can call to get technical support for the Web-IO or for the overall project

Internet
Homepage of the company responsible for the device.

Device Type
Company-specific device name

Device No
Company-specific article number.

Logo
Path and name of the logo positioned above the navigation tree. If you specify only logo.gif, the Web-IO uses the internally stored logo. Alternately you can reference a JPG or GIF file located on another server.

Note that the dimensions of the logo must closely approximate those of the W&T logo (width max. 200 pixels, height max. 100 pixels) to retain clarity in the navigation frame.

Send the settings to the Web-IO by clicking on the *Temporary Storage* button.

Activate the new settings under *Config >> Session Control >> LogOut, Save.*

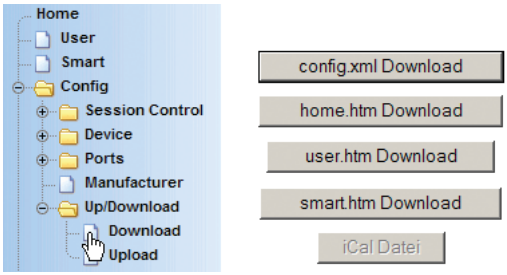
6.6 Upload/Download

The Web-IO Digital allows the user to upload or download custom HTML pages, configuration data and the logo.

6.6.1 Download

To download an HTMP page or the configuration, select in the navigation tree *Config >> Up/Download >> Download:*

Required access rights: *Administrator*



Config.xml Download
The entire Web-IO configuration is stored in the unit as an XML file. This means that when there are projects involving multiple Web-IOs having the same configuration, the Administrator can make configure just one unit and load the settings into all the others.

Start the download of the XML file by clicking on the corresponding button.

The XML parameters are then displayed in the browser window. To safe the XML data as a file, most browsers allow you to right-click the source text in the text editor and from there save it.

home.html / user.html Download
Here again the selected page is displayed first in the browser window and can be saved from the source text display. For a detailed description of this procedure, refer to the section *HTTP*.

iCal file
To configure the Web-IO as a time switch, the currently active iCal files can be downloaded here.

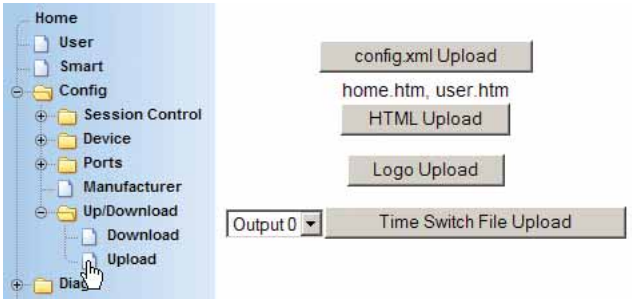
Details about time-based switching can be found in section „5.3 Time Switch - Web-IO as time switch“

6.6.2 Upload

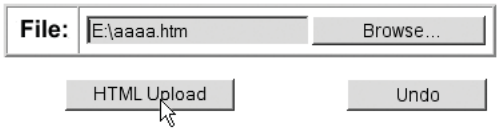
Just as you can download HTML and XML files, it is also possible to upload them to the Web-IO.

Select *Config >> Up/Download >> Upload:*

Required access rights: *Administrator*



config.xml Upload
The entire configuration of the Web-IO can be loaded into the Web-IO as an XML file in one operation. Clicking on the corresponding button displays an entry field in the configuration frame in which you can enter or select the upload file:



Click on the *Upload* button to start the upload.

A convenient way of saving, managing and later uploading XML files from the Web-IO is by using the Wutility tool.

At our Web site www.WuT.de you will find at left the Insider Search for Article No. Enter here article no. 57630, select „Tools“ in the field below, and click on „Go“. On the page which then opens select the link to the Wutility tool.

HTML Upload
The upload procedure for the home.htm and user.htm page is exactly analogous to the XML upload.

The exact procedure is described in detaili in the *HTTP* section.

logo Upload
For a custom appearance of the Web-IO start page, you can replace the W&T logo in the navigation frame with another one of your choice (GIF or JPG).

Clicking on the corresponding button displays an entry field in the configuration frame in which you can enter or select the upload file.

Click on the Upload button to start the upload.

For the uploads to remain in effect even after a power interruption, activate the new settings under *Config >> Session Control >> LogOut, Save. Time Switch File Upload*

Time Switch File Upload
Here you can upload the calendar data which determine the time-based switching in iCal format.

Details about time-based switching can be found in the section „5.3 Time Switch - Web-IO as time switch“

To ensure that the uploads are still in effect after power has been interrupted, enable the new settings using *Config >> Session Control >> LogOut, Save.*

6.7 Diagnostics and testing

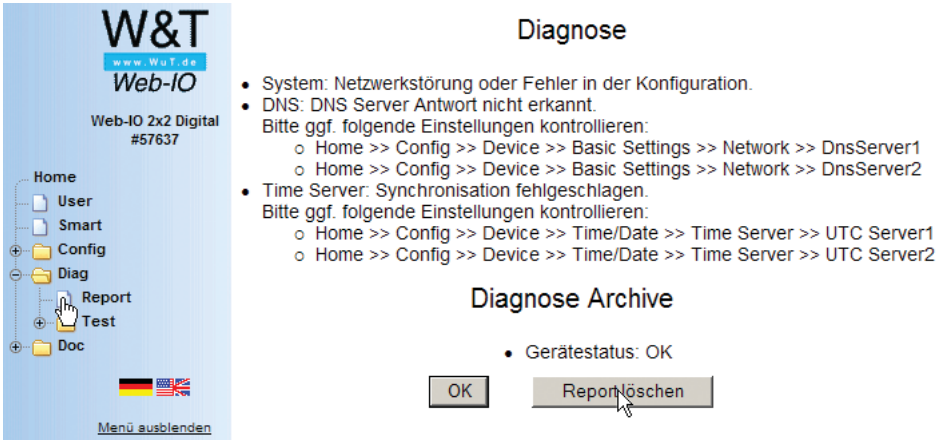
The Web-IO provides internal error management and a diagnostics system.

6.7.1 Error report

When an error occurs, this is indicated on the device by the *on error* LED flashing. In addition, any error states occuring are documented in a report and can be read out later.

All error messages are stored in the Web-IO and are retained even if the cause of the error has been remedied.

To do this, select *Diag >> Report:*



Clicking on the *Delete report* button allows the Administrator to clear all error messages.

m Required access rights: *Administrator*

A reset also deletes the report, regardless of whether it is performed by pressing the Reset button on the device,

interrupting the supply voltage or by using the Reset function from the Logout page.

An exact list of possible error messages and their cause can be found in the Appendix. In the section „Socket programming“ you are shown how errors can also be processed from your own applications.

You can also process error and diagnostic messages using SNMP traps or as a Syslog system message. See the SNMP and Syslog sections.

6.7.2 Configuration

Overall view

The Web-IO allows the Administrator to view and check the Web-IO configuration on a comprehensive Web page.

Select *Diag >> Test >> Check Config*

Validation of configuration

Parameter	HTTP	UDP	SNMP	Mail	Syslog	FTP	Binary 1	Binary 2	Logic	Time Switch
Enable Flag	—	OFF	ON	OFF	OFF	OFF	ON	OFF	—	—
Output Mode	OK	OFF	OFF	—	—	—	OFF	OFF	OFF	OFF
Source Port	80	42279	161	auto	auto	auto	—	49154	—	—
Source IpAddr	10.40.22.126	10.40.22.126	10.40.22.126	10.40.22.126	10.40.22.126	10.40.22.126	10.40.22.126	10.40.22.126	—	—
Destination Port	n.a.	n.a.	—	—	—	—	—	—	n.a.	n.a.
Destination IpAddr	—	—	—	—	—	—	—	—	—	—
Logic Funcs	—	—	—	—	—	—	—	—	OFF	—
Active	**ON**	OFF	**ON**	OFF	OFF	OFF	FAIL	OFF	OFF	OFF
Profile	Browser, ASCII Kommandostrings	ASCII Kommandostrings per UDP	SNMP	Mail	Syslog	FTP	Box-2-Box, Binärer Socketzugriff, OPC		Logik	Time Switch

Parameter	Alarm Mail	SNMP Trap	UDP Client	Pending TCP	TCP Client	Syslog Message	FTP Message
Alarm / Trap	OFF	OFF	OFF	OFF	OFF	OFF	OFF

Connection Recources	
Mode	**Max**
Pending TCP	3

The Web page shows which access types are activated with which parameters. Here the Web-IO performs a plausibility check of the settings. If missing parameters are detected which prevent proper operation of the access type, the corresponding fields are highlighted in orange.

Also checked and displayed are which transmission channels are selected for the alarms and whether all necessary parameters have been configured. Here again the access types which were not fully configured are highlighted in orange.

Finally, the number of possible static connections on Port 80 of the Web-IO is checked and displayed. Port 80 of the Web-IO can be used both for opening the Web pages as well as for socket applications.

The factory default setting is for all free ports configured so that they work as a server port on Port 80. Dynamic connections are permitted on only two ports.

If for example 5 ports are configured on HTTP (Port 80), 3 of them can be used by continuous TCP connections, e.g., by socket applications. The remaining two free ports can accept connections, but the Web-IO rejects then again after a short timeout.

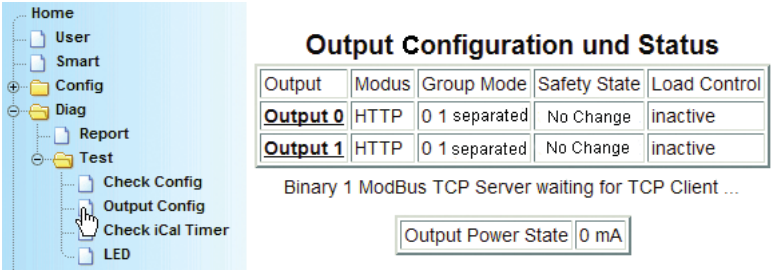
This ensures that you can always reach the Web-IO via the browser.

Overview of the output configuration

In addition to the overview of the general configuration, the Web-IO also offers an additional Web page which shows the configuration and status of the outputs.

Select *Diag >> Test >> Output Config*

m Required access rights: *Administrator*



The following information is output for each of the outputs:

Mode
Displays the configured mode

- HTTP
- UDP ASCII
- SNMP
- Binary 1
- Binary 2
- Logic

Group Mode
If 2 or 4 outputs are combined in a group, that is indicated in this column.

Safety State
This column may contain the following entries:

- *unchanged*
  No watchdog function was set for this output.
- *ON*
  A watchdog function was set for this output.
- *Error*
  When Load Control is activated an overload or the absence of a load is detected.
  In case of overload the Web-IO turns the output off.
  (not for Web-IO models with relay outputs)

Since the latter case may represent a serious error, these fields are highlighted in red.

Load Control

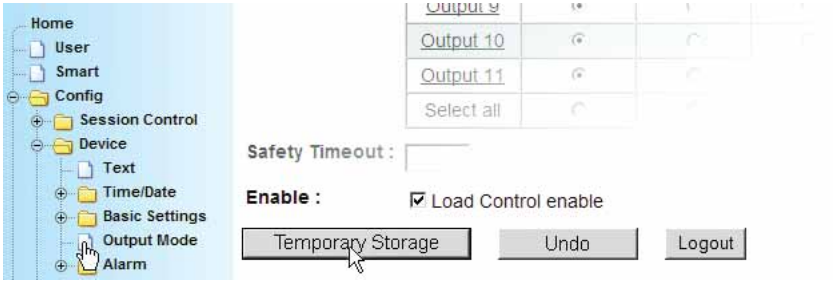
(not applicable to Web-IO with relay outputs)

In addition to the thermal short circuit protection, the Web-IO 12xDigital has load monitoring which in case of error turns the outputs off until they are enabled again by the Administrator.

The factory default setting is for Load Control not activated.

Under *Config >> Device >> Output Mode* you can activate or deactivate load monitoring by selecting *Load Control Enable.*

m Required access rights: *Administrator*



Regardless of whether Load Control is activated or not, the thermal overload protection protects the affected output from damage.

Activating Load Control enables a management function which in case of overload can turn the affected outputs OFF and notify the user.

For hardware reasons, load monitoring is done in pairs, i.e., always for two outputs at a time, for example Output 0 and Output 1. Depending on the state of the outputs and the error type, the Web-IO may not be able to definitively determine which output is responsible for the error. In this case both outputs are set to OFF.

Also monitored is whether any load at all is present. If the Web-IO can definitively determine that the error represents the absence of a load, Load Control is deactivated (only) for this

pair of outputs. By clicking on the *Delete report* button under *Diag >>Report* the Administrator can turn the outputs back on.

Required access rights: *Administrator*

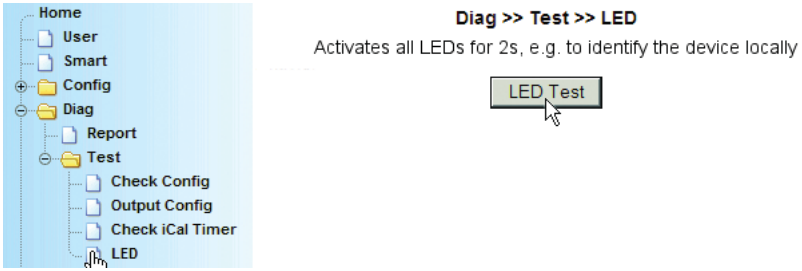
In the case of error messages generated by Load control, the physical wiring of the affected outputs must in any case be checked before they can be enabled again.

Overload errors can also be configured as triggers for an alarm. See section on Alarms.

6.7.3 LED test

To test all LEDs for the input/output section of the Web-IO 12xDIgital, select *Diag >> Test >> LED* and click on the *LED Test* button.

Required access rights: *Default User*, *Config*, *Administrator*.



All LEDs are turned on for approx. 2 sec.

This function can be used to uniquely identify a Web-IO in the control cabinet.

The actual state of the outputs is thereby not changed. The 3 LEDs for the Ethernet connection are not affected by this test.

7 Appendix

- IP address assignment alternatives
- Firmware update
- Emergency access
- Port numbers and network security
- Timing and signal processing limits
- Error messages and their causes
- Technical data
- Index

7.1 IP address assignment alternatives

In most cases in which the IP address is not assigned using DHCP, the simplest method is to use the WuTility tool for assigning an IP address to the Web-IO (see section 2.2).

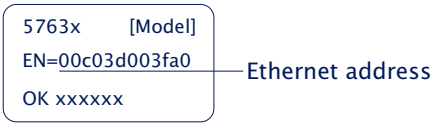
In exceptional cases or if no PC with a Windows operating system is available the following alternatives are available:

- Using the ARP command
- Using the serial port
- Using RARP protocol
- Using DHCP protocol
- Using BOOTP protocol

7.1.1 Assigning the IP address using the ARP command

This method can only be used if the Web-IO does not yet have an IP address, i.e., if the entry is 0.0.0.0. To change an IP address, use one of the other methods described in this section or use the menu item Config/Device/Network in the Web-Based Management of the Web-IO.

An additional prerequisite is a computer which is located in the same network segment as the Web-IO and on which TCP/IP protocol is installed. Read off the Ethernet address of the Web-IO from the sticker on the side of the housing:



Now use the following command to add a static entry to the ARP table of your PC :

```
arp-s[IPaddress] [MACaddress]
```

Example under Windows:

```
arp -s 10.40.72.15 00-C0-3D-00-3F-A0
```

Example under SCO UNIX:

```
arp -s 10.40.72.15 00:C0:3D:00:3F:A0
```

Older Windows systems accept a static entry only if a dynamic entry already exists. Here you should first ping another network station.

Finally start the Web browser and enter the following URL:

http://[IP address] [*Return*]

Example:

In Windows environments IP addresses must be entered without leading zeroes. Otherwise the system interprets the entry incorrectly and an incorrect IP address is assigned to the Web-IO.

The Web-IO takes the IP address of the first network packet sent to its Ethernet address as its own and saves it in non-volatile memory. The HTTP connection is then opened and the Home-page of the Web-IO appears in the browser. All other settings can now be easily made via Web-Based Management (see sec*tion Basic setting of the network parameters*).

7.1.2 Assigning the IP address through the serial port

After a reset of the Web-IO a time window of approx. 1-2 seconds is made available, during which entering at least 3 „x“ makes it possible to assign a new IP address.

In contrast to the previously described method via ARP, the serial way works regardless of whether the

Web-IO already has an IP address or not. The procedure may be repeated as often as desired. Therefore you should use this method if you don't know the IP address or have forgotten it. The appendix to this manual contains a detailed description for using Windows with Hyperterminal.

First connect the Web-IO to the serial port of a computer. For a standard PC a *crossed* RS232 cable (= null modem cable) is required (see section *Serial connection*).

The serial transmission parameters of the terminal program used are configured for *9600 baud*, *no parity*, *8 bits*, *1 stop bit*, *no handshake*. Press the reset button on the Web-IO briefly and enter the letter „x" at least three times on the terminal until the prompt *IPno.+<Enter>:* appears in the terminal program.

Use the conventional format (xxx.xxx.xxx.xxx) to enter the IP address, and finish your entry with <*Enter*>. If the entry was accepted, the system acknowledges with the assigned IP address. Otherwise a *FAIL* message is issued followed by the last current IP address.

All additional settings can now be made easily via Web-Based Management in the browser (see section *Basic setting of the network parameters*).

Serial assignment of subnet mask and gateway

Together with the IP address, the subnet mask and gateway address can also be assigned serially. The entry is comma delineated following the IP address. The following example assigns IP address 10.40.72.15, subnet mask 255.255.0.0 and gateway 10.40.1.1 to the Web-IO.

```
IP no.+<ENTER>:                      <- Web-IO
10.40.72.15,255.255.0.0,10.40.1.1    -> Web-IO
10.40.72.15 ,255.255.0.0,10.40.1.1-1 <-Web-IO
```

Serial deactivation of the DHCP-/BOOTP client

The DHCP-/BOOTP function of the Web-IO can be turned off while serially assigning the IP address. We recommend doing

this in any case where the IP assignment will not be expressly performed using DHCP/BOOTP.

To deactivate the DHCP/BOOTP client, enter the option „-0“ (zero) immediately following the IP address (no spaces!) and finish the entry with <*Enter*>.

Example:

```
xxx                                    -> Web-IO
IP no.+<ENTER>:                        <- Web-IO
10.40.72.15,255.255.0.0,10.40.1.1-0    -> Web-IO
10.40.72.15,255.255.0.0,10.40.1.1-0    <-Web-IO
```

This of course functions the same way when entering the subnet mask and gateway..

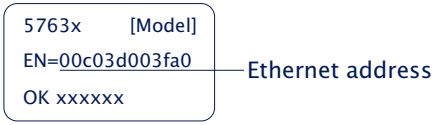
You can reactivate the function at any time later by using Web-Based Management im the browser.

Menu item: *Config >> Device >> Network*

7.1.3 IP assignment via RARP server

UNIX environments especially often use RARP protocol for centrally assigning IP addresses. Here TCP/IP devices that want an IP address send RARP requests with your Ethernet address as a network broadcast.

Activate the RARP server on the UNIX system and enter the Ethernet address of the Web-IO in the file */etc/ethers*. You must also enter the IP address of the Web-IO in the file */etc/hosts*.



The Web-IO must be located in the same subnet as the RARP server.

Example
Your Web-IO has the MAC address EN= 00C03D003FA0 (device sticker). You want to give it the IP address 172.16.232.17 and the alias name WEBIO_1:

- Entry in the file */etc/hosts*:
  `10.40.72.15 WEBIO_1`
- Entry in the file */etc/ethers*:
  `00:C0:3D:00:3F:A0 WEBIO_1`

If the Web-IO's IP address is 0.0.0.0 (=Factory Default), RARP broadcasts are cyclically generated in order to obtain a valid address from any existing RARP daemon.

If the Web-IO already has a valid IP address, an RARP broadcast is generated following each reset. If a reply follows within 500ms, the IP address contained therein is accepted. As with assigning via the serial port, this method also makes it possible to overwrite a current IP address.

7.1.4 Assigning the IP using DHCP protocol

Many networks use DHCP (Dynamic Host Configuration Protocol) or its predecessor described in the following section for centralized and dynamic assignment of the network parameters. DHCP protocol is activated by the factory default settings, so that in network environments dynamic IP assignment is sufficient for connecting the Web-IO to the network. The following parameters can be assigned using DHCP:

· IP address
· Subnet mask
· Gateway address
· DNS server
· Lease time

To prevent unintended address assignments or changes, we recommend deactivating the DHCP, BOOTP and RARP protocols if they are not expressly used in the respective

network environment. Web-IOss with incorrectly assigned IP addresses can be easily found after the fact using the scan function of the WuTility management tool and reconfigured.

Activating/Deactivating DHCP

The factory default setting is for DHCP protocol active. To deactivate it or to enable it again later, use one of the following options.

- **Management-Tool WuTility**

Select the desired Web-IO in the device list and click on the IP Address button. In the first dialog box enter the newly assigned network parameters and then click on Continue. In the following dialog BOX deactivate the options BOOTP and DHCP. Clicking on Continue then sends the new configuration data to the Web-IO.

- **Serial port**

As part of serial IP assignment, the following options for deactivating/activating DHCP and BOOTP can be selected directly following the address string:

-0 → Deactivates DHCP and BOOTP
-1 → Activates BOOTP/RARP
-2 → Activates DHCP

A detailed description of the procedure can be found in the section on *Serial assignment of IP, subnet mask and gateway*.

- **Web Based Management**

In the menu branch *Config >> Device >> Basic Settings >> Network -> DHCP/BOOTP Enable* you can alternatingly enable or disable both protocols. For detailed information refer to the section *Configuring the network parameters.*

System Name

To support any automatic updating of the DNS system by the DHCP server, the Web-IO identifies itself within the DHCP protocol with its system name. The factory default setting for this is *WEBIO_* followed by the last three places of the Ethernet address. For example the factory set system name of a Web-IO with the Ethernet address 00:c0:3d:01:02:03 is *WEBIO_010203*.

The system name of the Web-IO can be changed in the configuration. For additional information refer to the section *Menu: Config >> Device >> Text*

Lease-Time
The lease time determined and transmitted by the DHCP server specifies the Time-To-Live of the assigned IP address. After half the lease time has expired, the Web-IO attempts to extend the time for the assigned DHCP server and up update the address. If this is not possible by the time the lease time expires, for example because the DHCP server can no longer be reached, the Web-IO deletes the IP address and starts a new cyclical search for alternate DHCP servers for the purpose of assigning a new IP address.

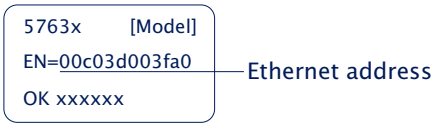
Because of the absent clock, the lease time associated with the current IP address is no longer available after a reset. After the restart therefore a corresponding update request is issued with the original DHCP server. If the latter is not resolvable at this point in time, the Com-Server deletes the IP address and starts a new cyclical search for alternate DHCP servers.

If DHCP is activated, the remaining lease time together with the current IP address is displayed in the menu item *Config >> Device >> Basic Settings >> Network* using the format hh:mm:ss.

If after the assigned lease time has expired the DHCP server is not reachable, the Web-IO deletes its IP address. All existing TCP/UDP connections between the Web-IO and other network clients are thereby closed. To prevent such events, we recommend configuring the assigned lease time in the DHCP server to infinite if possible.

Reserved IP addresses
If the Web-IO is used as a TCP server or UDP peer, it provides services which other clients in the network can also make use of as needed. To open a connection, they of course need the current IP address for the Web-IO, so that in such situations it makes sense to reserve a particular IP address for the Web-IO on the DHCP server. This is generally done by linking the IP

address to the unique Ethernet address of the Web-IO, which can be found on the sticker attached to the housing.



Dynamic IP addresses
Fully dynamic address assignment, whereby the Web-IO gets a different IP address every time it is restarted or after the lease time has expired, only makes sense in network environments with automatic cross-connection between the DHCP and DNS services. This means when a new IP address is assigned to the Com-Server, the DHCP server then automatically updates the DNS system as well. The new address is associated with the respective domain name. If in doubt, consult your system administrator for detailed information about your network environment.

If the Web-IO is configured as a TCP or UDP client and itself actively searches for a connection to server services in your network, dynamic changing IP addresses may be used.

7.1.5 Assigning the IP using BOOTP protocol

Some networks use BOOTP as predecessor of DHCP protocol for centralized and dynamic assignment of IP addresses. The factory default setting is for BOOTP turned off. You can activate it from *Config >> Device >> Basic Settings >> Network*. The following parameters can be assigned:

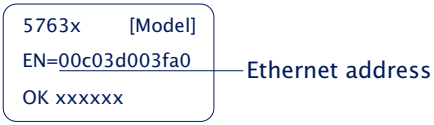
- IP address
- Subnet mask
- Gateway address
- DNS server

To prevent unintended address assignments or changes, we recommend deactivating the DHCP, BOOTP and RARP protocols if they are not expressly used in the respective network environment. Web-IOs with incorrectly assigned IP

addresses can be easily found after the fact using the scan function of the WuTility management tool and reconfigured.

Address reservation

BOOTP protocol is based on fixed reservations of fixed IP addresses for particular Ethernet addresses. This means a Web-IO connected to the network only gets an IP address if the latter was previously stored in the BOOTP server. Check with your system administrator for creating this reservation. The Ethernet address of the Web-IO can be found on the housing sticker.



Once the administrator has made the necessary entries, the Web-IO obtains the desired IP address automatically after each reset. To ensure accessibility of the Web-IO even should the BOOTP server go down, the previous IP address is retained should there be no reply.

7.2 Firmware Update

The operating software of the Web-IO is under constant development. The following section describes how to perform an upload for the latest firmware version.

7.2.1 Where can I get the current firmware?

On our web site www.WuT.de, having selected the Englisch language version, on the left side you find „Insider search via article number". Enter here the article no. 57630, select „Firmware" in the checkbox below and click „Go". On the page that will open, you select the link to the latest firmware.

7.2.2 Firmware update over the network using WIndows

For this you will need a PC running under Windows 9x/NT/2000/XP with a network connection and activated TCP/IP stack. For the update process you will need two files which, as already described, are available from the Homepage at *http://www.wut.de*.

- The executable *WuTility* -Tool for sending the firmware to the Web-IO
- The file with the new firmware you want to send to the Web-IO

No special preparation of the Web-IO is necessary for performing the firmware update.

The *WuTility* used for the update recognizes all W&T devices located in your network and is for the most part self-explanatory. If you do have questions or anything is unclear, please refer to the associated documentation or use our online help.

Be sure to use only the current version of Wutility for the update. Use of older Wutility versions may render the Web-IO inoperable.

We recommend deleting all passwords and performing a Reset of the Web-IO before the update is loaded.

For the update Wutility requires unrestricted network access to the Web-IO. Software firewalls and Internet security tools should be configured so that access to the Web-IO is not prevented.

Never intentionally interrupt the update process by disconnecting power or actuating the Reset button. After an incomplete update the Web-IO will be non-operational.

Never mix files with different version numbers in the file name. This will render the device inoperable.

The Web-IO normally recognizes when transmission of the new operating software is complete and then automatically performs a reset. In rare cases it may be necessary after an update to restart the Web-IO using a hardware reset (Reset button of Power ON). A software reset via network access through Port 8888 will not then be sufficient.

Web-IOs which before the update have a firmware version 1.34/1.00 or older can only have their firmware updated to the newest version by W&T (the unit must be returned).

Web-IOs which before the update have a firmware version 1.47/2.09 or older should be reset to factory defaults after the update.

7.3 Emergency access

The serial port (DTE) of the device provides an emergency access. Activiate this as follows:

Use a serial cable to connect the device to a PC (null modem connection) and start a serial terminal program. Use the following program settings:

9600,8,N,1,no handshake

Press the Reset button on the device. As soon as you hear the tone, press the following letters on your keyboard 3 times for the individual accesses:

3x „u“ Opens the update port. You can now load a firmware update

3x „f“ Resets the device to its factory default settings.
All previous configuration settings (including the IP address) are lost.

3x „p“ Deletes all previously assigned passwords.

For confirmation the **system error** and the **on error http** LEDs flash several times in succession.

3x „x“(Enter directly after pressing the Reset button, before the audible tone) assigning/changing the IP address. At the prompt, enter the desired IP address.

7.4 Ports and network security

7.4.1 Ports

Following is an overview of all the ports used in the Web-IO Digital.

Each port number may be used for only one service in the Web-IO. If differing numbers are used with the changeable ports, be sure that no port number is used twice.

7.4.2 The W&T Web-IO and network security

Security in networks is justifiably gaining increased attention. All the experts agree that there can be no such thing as abso-

lute security at today's state of technology. Each customer must therefore find an appropriate balance between security, functionality and cost for his specific requirements.

To give the customer the greatest possible degree of flexibility in this regard while responding to changing security requirements from a pure testing and installation environment to critical production applications, the security measures offer a high degree of user configurability.

This document provides an overview of the security measures implemented on the Wiesemann & Theis Web-IO and how they can be used. It is assumed that the original W&T firmware (without custom modifications) is used. For additional details, refer to the respective sections in this Manual.

7.4.3 The authorization concept of the Web-IO:

As already described in the section Web-Based Management, the Web-IO uses three authorization levels:

- Default User (with no special login)
- Operator
- Admin

These users normally access Port 80 through the Web interface, as is customary with http.

The default user can use the Web interface to read Web-IO data, and has access to the diagnostic tools as well as some online documentation (abbreviated manual, data sheet for the device, support information).

The operator can also set the outputs and configure the alarm outputs. No other settings can be changed by the operator.

The Administrator has full access to all the functionalities of the device. He specifies the operation mode(s) used and the network parameters.

Passwords can be defined for the *Operator* and *Admin*. The device does not require that passwords be used, nor that they must be different for the *Operator* and *Admin*, nor the minimum length, composition or regular changing of othe passwords. This is left to the user as he sees fit.

No encoding is used to send the password to the Web-IO. You should therefore ensure that password-protected access is only over an Intranet which the customer considers to be secure. For accesses over the public Internet, additional measures such as a VPN tunner (Virtual Private Network) may need to be taken. But this is a general problem of network security, and every customer needs to find his own solutions.

7.4.4 Ports with special function:

In addition to access over the Web interface, a variety of functions can be activated through various TCP or UDP ports. These are shown in Table 1 (Excel table). A series of functions can be turned on and off, but only with „Admin“ rights. The basic recommendation is to turn off all the functions you are not using. More on this subject can be found in the Manual.

SNMP:
To be able to incorporate the Web-IO into an SNMP-based systems management, all the essential configuration settings can also be modified via SNMP. Access is protected by using the Admin password in SNMP as a *community string*. SNMP access can be turned off.

Ports for inventorizing using the „WuTility“ tool:
Like all the „intelligent“ components from W&T, the Web-IO can be accessed using the „WuTility“ tool. The tool can use ports 8512 and 8513 to read information such as hard- and firmware version, IP address, etc. This access cannot be turned off, but write access to the device is not possible with the tool.

Firmware-Update:
The TFTP (TCP port 69) protocol makes it possible to get a firmware update and overwrite all the configuration settings.

Only, however, if update mode was previously enabled. There are several ways to do this:

- One-time activation using a special http string in connection with the Admin password (if set)
- One-time activation using *Config >> Session Control >> Log-out* and the *Firmware Update* button

Reset using Port 8888:
Opening a connection (e.g., using Telnet) to Port 8888 allows the device to be reset. Here you must give the Admin password (if set) right after the connection is opened. This immediate sending of the password is impractical by manual means, and should be performed by a program. The consequences of a reset are the same as for a momentary interruption of power or pressing the Reset button on the device.

7.4.5 Restoring the factory defaults

Restoring the factory faults allows you to undo all the security settings made previously. Among other things the Admin user is no longer password protected. This option thus represents also an „emergency access“ for cases where the password has been forgotten or in emergencies where the persons who know the passwords are not available.

Admin authorization is required to reset the factory defaults via software over the network. The factory defaults can however be restored through the serial port of the Web-IO by pressing a key combination after powering up or after a reset.

In any case you should be sure that only authorized persons have physical access to the unit including the serial port, which may mean keeping the device in a lockable control cabinet.

7.5 Timing and signal processing limits

The system architecture of the Web-IO tries to use the least possible energy to achieve the greatest possible processing speed. To accomplish this the Web-IO uses multithreading, a process whereby all the used services and functions share the overall system power.

7.5.1 Signal detection

Signal detection on the inputs has an especially high priority. Up to 500 signal changes per second are reliably detected and counted. The signals must be present for at least 1 ms (in short pulse mode 50µs for Inputs 0-3).

7.5.2 Signal transmission

Signal transmission over the network interface is asynchronous to signal detection. There are various reasons for this: For one thing, depending on the network load the entire network bandwidth may not always be available, and for another the ratio of protocol overhead to user data varies (sending of signal states via e-mail is for example much more involved than sending via a simple TCP connection). Depending on which mode is selected the application processes of the Web-IO also take more or less time. The time response can therefore vary considerably depending on the mode and protocol.

7.5.3 Signal timing

Two factors are considered in signal transmission timing:

- **Signal distance**
  The signal distance is the minimum required time between two signal changes for both changes to be not only detected (counted) but also sent.

Latency time

The latency time specifies how much time from the occurrence of a signal change is required before the corresponding message arrives at the communication partner.

Access mode	Input detection time between signal changes	Output switching time between signal changes	typical latency
Socket Access Web-IO as TCP-Server (Binary)	50ms	30ms	30ms
Socket Access Web-IO as TCP-Client (Binary)	150ms	30ms	30ms
Socket Access Web-IO as UDP-Peer (Binary)	120ms	30ms	30ms
Socket Access Web-IO as TCP-Server (Command String)	600ms	30ms	30ms
Socket Access Web-IO as TCP-Client (Command String)	300ms	30ms	30ms
Socket Access Web-IO as UDP-Peer (Command String)	polling only	30ms	30ms
Box-To-Box	50ms	50ms	30ms
SNMP	polling only	100ms	30ms
Alarm via Email	20s	-	20s
Alarm via SNMP-Trap	10s	-	10s
Alarm via Syslog	10s	-	10s
Alarm via FTP	2s	-	2s
Alarm via TCP	2s	-	2s
Alarm via UDP	2s	-	2s
Logic	2ms	2ms	1ms

The times given here are reference values for exclusive access in a local, non-loaded network. Depending on the network environment and application these values may vary significantly - especially if multiple modes are used in parallel.

7.6 Network inventory via UDP

Larger installations especially often require inventorizing of the Web-IOs or also W&T Com-Servers currently located in the network. UDP Port 8513 is provided for this purpose. After receiving a network packete on this port, the Web-IO replies to the sender with an info packet. This contains information about the Web-IO and its network parameters.

Whether a packet was received via broadcast or directly addressed makes no difference in generating the info structure. The only requirement is that Port 8513 be used.

For reasons of backward compatibility the Web-IOs continue to support UDP info port 8512. For new applications, however, only Port 8513 should be used.

7.6.1 The info packet

Each info packet consists of 3-6 data structures: *BOX_VERSION* contains information about the respective model of the Web-IO or Com-Server and its firmware version. Network-relevant data such as MAC and IP address can be taken from the *BOX_DESCRIPT* structure. The 3.-6. structure *PORT_DESCRIPT* provides information about the sete operation mode (for W&T Com-Servers only) and the current connection status of the individual ports. The total length of an info packet is calculated as follows: *10 + 22 + (port_anz x 10) bytes*

```
#pragma pack(1)

typedef struct _BOX_VERSION           // ( 10 byte )
  {       unsigned int version;         // 0x10: 1.0 (Version of this structure)
          unsigned int sw_rev;        // z.B. 1.24 (0x1501)
          unsigned int hw_rev;        // C2_EURO        = 2.0 (0x0002)
                                    // C3_PC104         = 3.0 (0x0003)
                                    // C4_MINI          = 4.0 (0x0004)
                                    // C5_100BT         = 5.0 (0x0005)
                                    // C6_INDUSTRY      = 6.0 (0x0006)
                                    // C8_LOW_VOLTAGE   = 8.0 (0x0008)
                                    // C8_WEB-IO            = 8.1 (0x0108)
          unsigned int reserved[2];
  }BOX_VERSION;
```

```
typedef struct _BOX_DESCRIPT          // ( 22 byte)
  {       unsigned char mac_addr[6];    // MAC address of the Com-Server
          unsigned long IP_number;     // IP address of Com-Server
          unsigned long gateway;        // Gateway
          unsigned long subnet_mask;    // Subnet Mask
          unsigned int  MTU;           // MTU
          unsigned int  port_anz;       // Number of ports in the Com-Server
  }BOX_DESCRIPT;                     // (port_anz * 10 byte)

typedef struct _PORT_DESCRIPT         // ( 10 bytes )
  { union
          { unsigned int wState;
            struct _new_type
              { unsigned char port_type;   // 0x01 = serial port (Serial CPU-Port)
                                           // 0x02 = serial port (UART Port)
                                           // 0x03 = reserved
                                           // 0x04 = Digital I/O
                                           // 0x05 = Analog I/O (in development)
                                           // 0x06 = W&T Dual Port RAM
                unsigned char state;        // 0=free, 1=connect, 2=waits
              };
            };
          unsigned int  mode;           // 0x0001 = TCP-Client
                                        // 0x0002 = TELNET-Client
                                        // 0x0003 = FTP-Client
                                        // 0x0004 = Box2Box-Client (aktiv)
                                        // 0x0005 = UDP Send/ReceiveMode
                                        // 0x0006 = MULTI PORT (DPRAM, SERIAL PROTOCOL)
                                        // 0x0007 = SNMP-Agent
                                        // 0x0008 = Box2Box Server (passive)
                                        // 0x0010 = SLIP Router
                                        // 0x0020 = PPP Router (in development)
                                        // 0x0030 = Box2Box Bus System: Slave Box
                                        // 0x0040 = Box2Box Bus System: Master Box
          unsigned long remote_IP;      // if state == CONNECT, otherwise 0
          unsigned int  remote_port;    // if state == CONNECT, otherwise 0
  }PORT_DESCRIPT;

typedef struct _WT_INTERN3            // ** all parameters in Hostorder / Low Byte First **
  {       BOX_VERSION  bv;            // Port = UDP_BOX_INFO_8513
          BOX_DESCRIPT bd;
          PORT_DESCRIPT pd[ACT_PROCESS];
  } WT_INTERN3;

#pragma pack()
```

The structure PORT_DESCRIPT does not have to be evaluated for the Web-IO. It is intended only for information pertaining to W&T Com-Servers.

All variables of type integer *and* long *are represented in host order. This means the low bytes come first. The IP-Adresse 172.17.2.3 appears for example in byte sequence 3, 2, 17, 172.*

UDP packets are connectionless and unsecured datagrams. Especially when using with broadcasts, both your own request packet as well as the reply packet from the Com-Server can become lost. To be sure you have gotten all the Com-Servers installed in a subnet, the request broadcast should therefore be repeated as necessary.

7.7 Error messages and causes

The following list provides an overview of all the error messages that can be shown in the report under *Diag >> Report*.

Error no.	Error message
0	Device status: OK
1	System: Network fault or configuration error
2	DNS: Format error in the DNS query
3	DNS: Internal DNS Server error
4	DNS: Name does ot exist
5	DNS: DNS-query type not supported
6	DNS: DNS-query was rejected
7	DNS: DNS Server cannot be reached
8	DNS: DNS Server reply not recognized
9	DNS: Wrong IP address
10	DNS: No resolution possible
11	Mail: Server refuses access
12	Mail: Sender transmission failed
13	Mail: Recipient transmission failed
14	Mail: Change to data mode failes
15	Mail: Transmission of mail text failed
16	Mail: Closing of mail server connection failed
17	Mail: Mail Server not responding
18	Mail: IP address or name of mail server not entered
19	Mail: Sending of the mail was cancelled and is being repeated
20	Transmission error
21	Protocol error
22	System: No router was configured
23	System: Destination IP address not recognized
24	System: A network fault was detected (cable open or no link)
25	System: The COM-Server triggered a reset
26	Warnung: Watchdog Timer Reset triggered
27	UDP: UDP Port number invalid: use Port 42279
28	System: Checksum error Factory Defaults were loaded
29	Snmp: IP address or name of SNMP Manager not entered
30	UDP Alarm: IP address or name of UDP Server not entered
31	UDP Alarm: UDP server port not entered
32	System: Number of pulses exceeds 2000/sec: Pulse mode Input 0 stopped
33	System: Number of pulses exceeds 2000/sec: Pulse mode Input 1 stopped
34	System: Number of pulses exceeds 2000/sec: Pulse mode Input 2 stopped
35	System: Number of pulses exceeds 2000/sec: Pulse mode Input 3 stopped
36	Binary 2: No or invalid Destination Port
37	Binary 2: No or invalid Destination IP address
38	System: Watchdog Timer expired
39	Binary 1: No or invalid Destination Port

Error No.	Error message
40	Binary 1: No or invalid oftination IP address
41	System: Port number assigned more than once
42	Binary 2: Invalid local port number
43	Binary 1: Invalid local port number
44	Box2Box: Access by Slave rejected, please check Box2Box settings
45	Box2Box: Slave is in Box2Box-Master mode
46	Box2Box: Master could not be cleared, plase log out
47	System: Load turned off: Details under Diag>>Test>OutputConfig
48	System: Clear restart with report
49	Reserved port number: please note Manual appendix
50	Time Server: Synchronization failed
51	System: Data Overrun Error Data in input buffer were overwritten
52	TCP Client Alarm: IP address or name of TCP server not entered
53	TCP Client Alarm: Port of TCP server not entered
54	TCP Client Alarm: Server not reached
55	Syslog Alarm: IP address or name of Syslog server not entered
56	Syslog Alarm: Port of Syslog servers not entered
57	Syslog Alarm: Server not reached
58	Reserved port number: Setting changed to Port 80
59	Mail: IP address or name of POP3 server not entered
60	FTP Alarm: IP address or name of FTP server not entered
61	FTP Alarm: Control Port of FTP Server not entered
62	FTP Alarm: Data Port of FTP Server not entered
63	FTP Alarm: Data transfer error
64	FTP Alarm: No user entry found
65	FTP Alarm: No password entered
66	FTP Alarm: No account entry found
67	FTP Alarm: No file name entered
68	FTP Alarm: Server not responding
69	PPP Modem: DCD Line Error
70	PPP Modem: Not found
71	PPP Modem: Command Error
72	PPP Modem: Busy
73	PPP Modem: No Carrier
74	PPP Modem: No Dialtone
75	PPP Modem: No Answer
76	PPP: Time Out
77	PPP: Authentication Error
78	DYNDNS Error: Service was turned off.
79	HTTP Client: IP address or name of HTTP Server not entered
80	HTTP Client: Port of HTTP Server not entered
81	The following count pulses could not be associated with a direction: Counter x

7.8 Technical Data

7.8.1 Technical Data 57630/57631

Network:	Ethernet 10/100BaseT autosensing
Protocol:	TCP and UDP Sockets, Client and Server SNMP incl. Traps SMTP (E-Mail), FTP, Syslog OPC Server Inventory, group management
Latency times:	The Web-IO 12xDigital is designed for signal change times > 50ms. The time behavior is also heavily dependent on the network load and network infrastructure.
Digital outputs:	12 x Digital Out 6V-30V, 500mA Grouping à 2 or 4 outputs Max. group current 2A Max. total current 6A Minimum load 22kOhm Thermal short circuit protected
Digital inputs:	12 x Digital In, Max. input voltage +/-30V Reverse polarity protected within this range Switching threshold 8V +/- 1V "On" current = 2.2 mA
Terminations:	2 x 16 screw terminals
Galvanic isolation:	Digital outputs - Network: min. 500 V to network Digital inputs: min. 1000 V
Serial port:	Configuration port with RS232 interface DB9-plug, configured like PC-COM
Baud rate:	9600 baud
Data format:	8 data bits, 2 stop bits No Parity
Flow control:	No Handshake
Displays:	Status LEDs for network 24 LEDs for digital states
Operating elements:	1 hidden reset button
Supply voltage:	12-24V AC/DC
Storage temperature:	-25°C - 70°C
Operating temperature:	0°C - 60°C
Housing:	Plastic housing for top-hat rail mount 106.8 x 87.8 x 62.6 (l x w x h)
Weight:	approx. 260 g

7.8.2 Technical Data 57637, 57638

Network:	Ethernet 10/100BaseT autosensing
Protocol:	TCP and UDP Sockets, Client and Server SNMP incl. Traps SMTP (E-Mail), FTP, Syslog OPC Server Inventory, group management
Latency times:	The Web-IO 12xDigital is designed for signal change times > 50ms. The time behavior is also heavily dependent on the network load and network infrastructure.
Digital outputs:	2 x Digital Out 6V-30V, 500mA Grouping à 2 outputs Max. group current 1A Max. total current 1A Minimum load 22kOhm Thermal short circuit protected
Digital inputs:	2 x Digital In, Max. input voltage +/-30V Reverse polarity protected within this range Switching threshold 8V +/- 1V "On" current = 2.2 mA
Terminations:	2 + 6 screw terminals
Galvanic isolation:	Digital outputs - Network: min. 500 V to network Digital inputs: min. 1000 V
Displays:	Status LEDs for network and power
Supply voltage:	PoE or 24V - 48V AC/DC
Storage temperature:	-25°C - 70°C
Operating temperature:	0°C - 60°C
Housing:	Plastic housing for top-hat rail mount 105 x 75 x 22mm (l x w x h)
Weight:	approx. 260 g

7.8.3 Technical Data 57634

Network:	Ethernet 10/100BaseT autosensing
Protocol:	TCP and UDP Sockets, Client and Server SNMP incl. traps SMTP (E-Mail), FTP, Syslog OPC-Server Inventorizing, group management
Latency times:	The Web-IO 12xDigital is designed for signal change time > 50ms. The time response is also highly dependent on the network load and network infrastructure.
Digital outputs:	6 potential-free contacts (normally open) Switching voltage max. 48V AC at 5A load current Switching voltage max. 30V DC at 5A load current Pause between two switching operations min. 200ms
Digital inputs:	12 x Digital In, max. input voltage +/- 30V Reverse polarity protected within this range Switching threshold 8V _+/- 1V "On" current = 2.2 mA
Connections:	2 x 16x screw terminals
Galvanic isolation:	Digital outputs - network: min. 500V to network Digital inputs: min. 1000V
Serial port:	Configuration port with RS232 interface DB9 male, pinout for PC-COM
Baud rate:	9600 baud
Data format:	8 data bits, 2 stop bits No Parity
Flow control:	No Handshake
Displays:	Status LEDs Network 24 LEDs for digital states
Operating elements:	1 covered Reset button
Supply voltage:	12-24V AC/DC Current draw at 12V approx. 300mA Current draw at 24V approx. 150mA
Storage temperature:	-25°C - 70°C
Operating temperature:	0°C - 60°C
Housing:	Plastic housing for DIN rail mount 106.8 x 87.8 x 62.6 (l x w x h)
Weight:	approx. 260 g

7.9 Disposal

Web-IO devices are not to be disposed of along with normal waste, but rather at an official collection point for electronic waste.

Web-IO models 57634,57637 and 57638 contain a non-rechargeable lithium button cell type BR (Lithium Poly Carbon Monofluoride) for maintaining the clock time even when the device is turned off, and this battery must be disposed of separately when it is depleted.

Take it to an official battery recycling site.

To remove the battery, first disconnect the unit from all cables and sensors.

Loosen the four clamps on the housing side and open the housing.

Remove both the screws used to attach the two lower circuit boards and remove the bottom-most board.

On the back side of the center board you will find the button cell. Remove it from its holder and take it to the recycling site.